Stabilized Finite Elements in FUN3D

W. Kyle Anderson*
NASA Langley Research Center, Hampton, VA 23681, USA
James C. Newman †
University of Tennessee at Chattanooga, Chattanooga, TN 37419, USA
and Steve L. Karman ‡
Pointwise Inc., Fort Worth, TX 76104, USA

A Streamlined Upwind Petrov-Galerkin (SUPG) stabilized finite-element discretization has been implemented as a library into the FUN3D unstructured-grid flow solver. Motivation for the selection of this methodology is given, details of the implementation are provided, and the discretization for the interior scheme is verified for linear and quadratic elements by using the method of manufactured solutions. A methodology is also described for capturing shocks, and simulation results are compared to the finite-volume formulation that is currently the primary method employed for routine engineering applications. The finite-element methodology is demonstrated to be more accurate than the finite-volume technology, particularly on tetrahedral meshes where the solutions obtained using the finite-volume scheme can suffer from adverse effects caused by bias in the grid. Although no effort has been made to date to optimize computational efficiency, the finite-element scheme is competitive with the finite-volume scheme in terms of computer time to reach convergence.

I. Introduction

FUN3D is a finite-volume code developed at the NASA Langley Research Center for solving fluid-dynamic problems associated with the analysis and design of aerospace vehicles.[1–6] The code is widely distributed and used for aerospace and nonaerospace applications by hundreds of users throughout the government, industry, and academia. The underlying technology is comprised of a finite-volume spatial discretization using Roe's approximate Riemann solver[7] with the unknowns located at the vertices of the mesh, where the solution is advanced at each time step using an implicit scheme based on backward-Euler time differencing and a simple Gauss-Seidel iteration scheme. Although this code has been quite successful for many applications, there are several drawbacks where improvement is needed. First, although the extent of the stencil is fixed, data is required from two layers of nearby neighbors, thereby increasing the difficulty in obtaining, maintaining, and extending, an exact linearization of the residual. This capability is important for obtaining sensitivity derivatives for simulation-based design, which is a unique capability of the code. The difficulties associated with linearizing the residual also raise the burden for using strong solvers often needed to achieve iterative convergence of the analysis problems, which can also be critical for achieving converged adjoint solutions. Secondly, with the current technology, there is no immediate means for extension to higher-order accuracy that does not also further extend the stencil, thereby exacerbating the problem previously discussed. Finally, while accurate results can be obtained on tetrahedral meshes, more elements are typically required when compared to hexahedral or even mixed-element meshes. To address these issues, the purpose of the current work is to eliminate the requirement for the large stencil, provide a clear path for higher-order schemes, and to improve the accuracy for tetrahedral elements.

Because finite-element methods can achieve arbitrary-order discretization accuracy using only nearest-neighbor data structures, these methods immediately offer a clear solution to the first two problems. In

*Senior Research Scientist, Computational AeroSciences Branch, AIAA Associate Fellow.
†Professor, Mechanical Engineering Department, AIAA Associate Fellow.
‡Staff Specialist, AIAA Associate Fellow.

this context, the discontinuous-Galerkin method has received significant attention and has been aggressively developed in recent years.[8–15] The stabilized finite-element methods, which include the Streamlined Upwind Petrov-Galerkin (SUPG) scheme,[16,17] Galerkin least squares,[18] and variational multiscale methods,[19] provide alternate discretizations that offer advantages in many common scenarios. Specifically, for moderate orders of accuracy, stabilized finite-element methods require many fewer degrees-of-freedom than discontinuous-Galerkin methods for computations on the same mesh.[20,21] Similarly, for implicit time-advancement algorithms under the same assumptions, the discontinuous-Galerkin scheme requires substantially more nonzero entries in the linearization matrix.[20,21] As a specific example, for linear tetrahedral elements, the discontinuous-Galerkin schemes require approximately 25 times more degrees-of-freedom on a given mesh when compared to the Petrov-Galerkin method. Similarly, for implicit time-advancement algorithms under the same assumptions, the discontinuous-Galerkin scheme requires a factor of 30 times more non-zero entries in the linearization matrix. While some of the associated extra work can be mitigated by using hexahedral elements, a discontinuous-Galerkin formulation on these elements still requires almost 8 times as many degrees-of-freedom as a continuous-Galerkin approach for trilinear elements, and almost three times as many for quadratic elements. Similar relationships hold for prismatic elements, with over 11 times as many degrees-of-freedom for linear elements, and four times as many for quadratic elements.

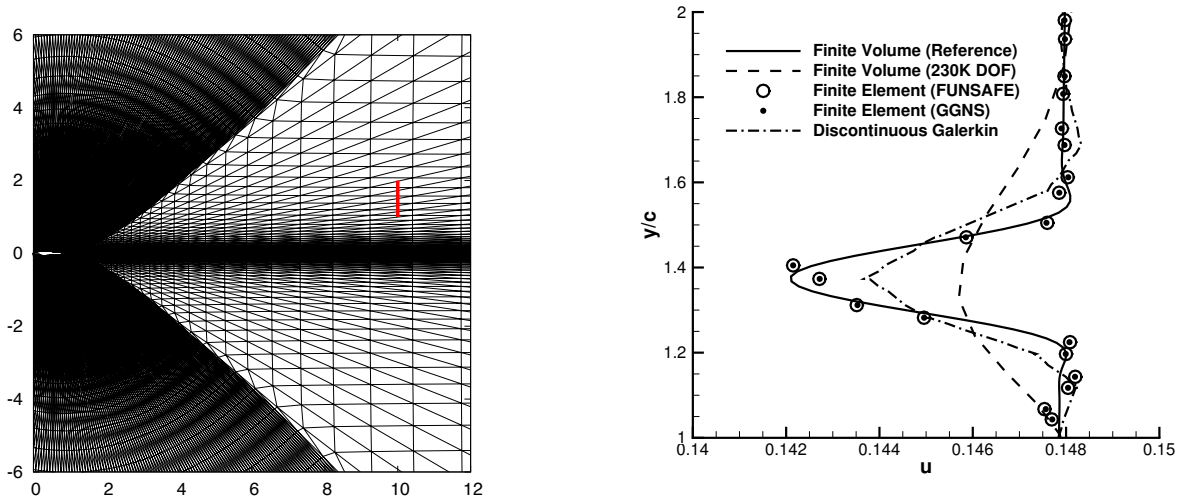
Because of the increased degrees-of-freedom associated with the discontinuous-Galerkin scheme, one would expect that increased accuracy would also be realized when compared to the continuous-Galerkin approach for computations on the same mesh. This assertion has been verified in Ref. 20 for Maxwell's equations, which are exactly analogous to the linearized Euler equations. This reference demonstrates that on the same mesh, the discontinuous-Galerkin and a SUPG Petrov-Galerkin method yield almost identical error levels when measured in the L_1 norm, whereas in the L_2 norm the errors from the discontinuous-Galerkin method are approximately thirty percent lower for both linear and quadratic elements. However, for the two-dimensional calculations considered in that section of Ref. 20, the discontinuous-Galerkin method requires six times more degrees-of-freedom for linear elements, and approximately three times as many degrees-of-freedom for quadratic elements, thereby not justifying the extra expense required for the moderate gain in accuracy. In three dimensions, the same reference reports that experiments with the discontinuous-Galerkin scheme are more expensive by a factor of 27 in comparison to the SUPG continuous-Galerkin method, whereas for quadratic elements, the discontinuous-Galerkin scheme is 12 times more expensive; these results agree well with estimates based simply on counting degrees-of-freedom. Subsequently, direct comparisons for turbulent Navier-Stokes simulations[22–25] have also demonstrated that while very similar results can be obtained using either methodology, a compelling rationale for favoring the discontinuous-Galerkin method over SUPG stabilized finite elements has not emerged using the moderate discretization orders considered.

Despite the advantages offered by the stabilized finite-element method for moderate orders of accuracy, discontinuous-Galerkin schemes remain far more popular for aerospace engineering simulations. In fact, an informal survey of research results[21] presented at AIAA conferences between 2009 and 2013 has indicated that research in discontinuous-Galerkin methods is reported more than ten times that of stabilized finite-elements. Although the stabilized finite-element community is comparatively small, notable progress in developing software based on this technology has been developed and applied to meaningful problems.[26–34]

From the above discussions, the stabilized finite-element approach appears to offer a favorable remedy to the first two problems associated with the finite-volume scheme; that is, the larger stencil and the lack of a viable means for extensions to higher order. In regards to the final criteria of increased accuracy, it remains to be demonstrate that the stabilized finite-element approach indeed meets these requirements.

To address this issue, two examples are cited to demonstrate that incorporating a SUPG stabilized finite-element method into FUN3D will likely provide the desired benefits. The first example is drawn from Ref. 35, which reports on results obtained as part of a workshop to examine accuracy of various discretization methods. These results, repeated in Fig. 1, show velocity profiles ten chord lengths downstream of an NACA 0012 airfoil at a freestream Mach number of 0.15, an angle of attack of 10°, and a Reynolds number, based on the chord of the airfoil, of 6.0×10^6. For comparison purposes, a benchmark solution had previously been established using the finite-volume scheme in FUN3D on a quadrilateral mesh with over 14 million nodes. Profiles, extracted along the line depicted in Fig. 1(a), are obtained using the FUNSAFE[20,25,35,36] stabilized finite-element scheme with linear basis functions on a mesh with only 230 thousand degrees-of-freedom. The profiles are plotted against the finite-volume solution, a discontinuous-Galerkin solution,[12,37] and an independently-developed stabilized finite-element method,[26] with all solutions obtained on the same mesh. The Riemann solver for the convective terms in the discontinuous-Galerkin formulation is a Roe-type

method that accounts for the convective coupling of the Reynolds-averaged Navier-Stokes equations and the turbulence model, whereas the viscous interface fluxes are formed using a symmetric interior penalty method.[12] With the exception of the reference solution, the other computations have all been performed on the same mesh, and that on this mesh, the discontinuous-Galerkin solution contains over one million degrees-of-freedom, whereas the other solutions only contain about 230,000. As seen in Fig. 1(b), the SUPG solutions demonstrate significantly better accuracy than those obtained using the finite-volume scheme, with the discontinuous-Galerkin results laying approximately midway between the two. The SUPG results obtained using linear elements, which have the same nominal order-of-accuracy as the finite-volume scheme, exhibit much less smearing of the profile and are able to capture the wake deficit with many fewer points. The fact that the stabilized finite-element results are quite good, despite being computed on triangular elements, gives credence to the claim that the SUPG scheme provides improved accuracy over the finite-volume scheme for this element type.



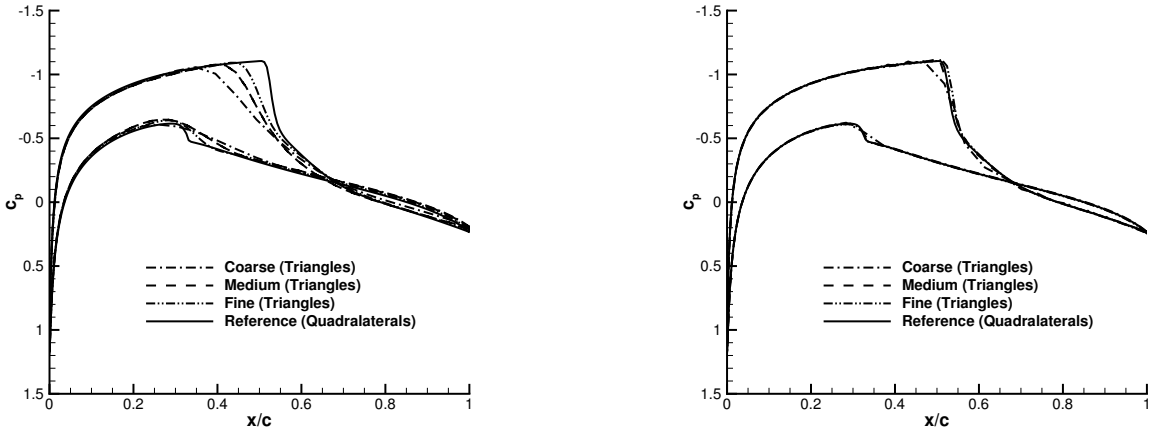
(a) Mesh and location of profiles.

(b) Profiles.

Figure 1. Profiles of u-component of velocity behind an NACA 0012 airfoil. $M_\infty = 0.15$, $\alpha = 10.0^\circ$, $Re = 6.0 \times 10^6$.

A second case is presented in Fig. 2 for the NACA 0012 airfoil, computed on the same series of meshes originally developed for the workshop previously mentioned. The freestream conditions for this case correspond to transonic flow, as evidenced by the shock wave located at approximately sixty percent chord on the upper surface of the airfoil. Here, a reference solution is again obtained using the finite-volume scheme with quadrilateral elements and comparisons are made with the finite-volume and finite-element schemes computed on triangular meshes. Specifically, the coarse-, medium-, and fine-mesh triangular grids contain 14,576, 57,824, and 230,336 nodes, respectively, whereas the reference quadrilateral mesh contains 919,428 nodes. As seen, the stabilized finite-element scheme again exhibits much less sensitivity to the mesh when compared to the finite-volume scheme, achieving mesh-independent shock positions with only 14,576 nodes.

The SUPG stabilized finite-element scheme can apparently address all three issues identified with the finite-volume discretization in FUN3D and, therefore, this scheme has been implemented as a library that can be compiled and linked with the main body of the flow solver. This work is ongoing and preliminary results have previously been reported in Ref. 28. Note that interest in the SUPG scheme among FUN3D developers dates back to the 1998 Ph.D. thesis of Bonhaus.[38] In that reference, two- and three-dimensional high-order results are obtained for inviscid flows, and laminar simulations are presented for two-dimensional flows. In the remaining sections of the paper, detailed descriptions of the new implementation for turbulent flow are provided, with results included to illustrate and evaluate the accuracy and performance of the scheme.



(a) Finite-volume solutions.

(b) Finite-element solutions.

Figure 2. Pressure distributions for NACA 0012 airfoil. $M_\infty = 0.798$, $\alpha = 1.44^\circ$, $Re = 3.0 \times 10^6$.

II. Governing Equations

The governing equations are the compressible, Reynolds-Averaged Navier-Stokes equations augmented with the one-equation Spalart-Allmaras turbulence model[39] that has been modified from the original model[40] to allow for negative values of the turbulence model working variable and will subsequently be denoted as the negative SA turbulence model. The equations can be expressed in the following conservative form:

$$\frac{\partial \mathbf{Q}(\mathbf{x},t)}{\partial t} + \nabla \cdot (\mathbf{F}_e(\mathbf{Q}) - \mathbf{F}_v(\mathbf{Q}, \nabla\mathbf{Q})) = \mathbf{S}(\mathbf{Q}, \nabla\mathbf{Q}) \quad \text{in} \quad \Omega \tag{1}$$

where Ω is a bounded domain. The vector of conservative flow variables $\mathbf{Q}$, the inviscid and viscous Cartesian flux vectors, $\mathbf{F}_e$ and $\mathbf{F}_v$, are defined by:

$$\mathbf{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho E \\ \rho\tilde{\nu} \end{bmatrix}, \quad \mathbf{F}_e^x = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ (\rho E + p)u \\ \rho u\tilde{\nu} \end{bmatrix}, \quad \mathbf{F}_e^y = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ (\rho E + p)v \\ \rho v\tilde{\nu} \end{bmatrix}, \quad \mathbf{F}_e^z = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ (\rho E + p)w \\ \rho w\tilde{\nu} \end{bmatrix} \tag{2}$$

$$\mathbf{F}_v^x = \begin{bmatrix} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} + \kappa\frac{\partial T}{\partial x} \\ \frac{1}{\sigma}\rho(\nu + \tilde{\nu}f_n)\frac{\partial \tilde{\nu}}{\partial x} \end{bmatrix}, \quad \mathbf{F}_v^y = \begin{bmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ \tau_{yz} \\ u\tau_{xy} + v\tau_{yy} + w\tau_{yz} + \kappa\frac{\partial T}{\partial y} \\ \frac{1}{\sigma}\rho(\nu + \tilde{\nu}f_n)\frac{\partial \tilde{\nu}}{\partial y} \end{bmatrix},$$

$$\mathbf{F}_v^z = \begin{bmatrix} 0 \\ \tau_{xz} \\ \tau_{yz} \\ \tau_{zz} \\ u\tau_{xz} + v\tau_{yz} + w\tau_{zz} + \kappa\frac{\partial T}{\partial z} \\ \frac{1}{\sigma}\rho(\nu + \tilde{\nu}f_n)\frac{\partial \tilde{\nu}}{\partial z} \end{bmatrix} \tag{3}$$

Here, ρ , p, and E denote the fluid density, pressure, and specific total energy per unit mass, respectively, $\mathbf{u} = (u, v, w)$ represents the Cartesian velocity vector, and $\tilde{\nu}$ represents the turbulence working variable in the negative SA model. The pressure p is determined by the equation of state for an ideal gas,

$$p = (\gamma - 1)\left(\rho E - \frac{1}{2}\rho(u^2 + v^2 + w^2)\right) \tag{4}$$

where γ is the ratio of specific heats, which is 1.4 for air. The subscripts on τ represent the components of the viscous stress tensor, which is defined for a Newtonian fluid as,

$$\tau_{ij} = (\mu + \mu_T)\left(\frac{\partial \mathbf{u}_i}{\partial \mathbf{x}_j} + \frac{\partial \mathbf{u}_j}{\partial \mathbf{x}_i} - \frac{2}{3}\frac{\partial \mathbf{u}_k}{\partial \mathbf{x}_k}\delta_{ij}\right) \tag{5}$$

where δ_{ij} is the Kronecker delta and subscripts i, j, k refer to the Cartesian coordinate components for $\mathbf{x} = (x, y, z)$. μ refers to the fluid dynamic viscosity and is obtained via Sutherland's law.[41] In Eq. (5), μ_T denotes the turbulence eddy viscosity, which is obtained from the negative SA model by:

$$\mu_T = \begin{cases} \rho\tilde{\nu} f_{v1} & \text{if } \tilde{\nu} \geq 0 \\ 0 & \text{if } \tilde{\nu} < 0 \end{cases} \tag{6}$$

The source term, $\mathbf{S}$, in Eq. (1) is given by $\mathbf{S} = [0, 0, 0, 0, 0, S_t]^T$, where the components for the continuity, momentum and energy equations are zero. The source term corresponding to the turbulence model equation takes the following form:[39]

$$S_t = P - D + \frac{1}{\sigma}c_{b2}\rho\nabla\tilde{\nu}\cdot\nabla\tilde{\nu} - \frac{1}{\sigma}(\nu + \tilde{\nu}f_n)\nabla\rho\cdot\nabla\tilde{\nu} \tag{7}$$

where the production term is given as

$$P = \begin{cases} c_{b1}\rho(1 - f_{t2})\tilde{S}\tilde{\nu} & \text{if } \tilde{\nu} \geq 0 \\ c_{b1}\rho(1 - c_{t3})S\tilde{\nu} & \text{if } \tilde{\nu} < 0 \end{cases} \tag{8}$$

and the destruction term is defined as

$$D = \begin{cases} \rho(c_{w1}f_w - \frac{c_{b1}}{\kappa_t^2}f_{t2})(\frac{\tilde{\nu}}{d})^2 & \text{if } \tilde{\nu} \geq 0 \\ -\rho c_{w_1}(\frac{\tilde{\nu}}{d})^2 & \text{if } \tilde{\nu} < 0 \end{cases} \tag{9}$$

In Eq. (7), (8), and (9), ν denotes kinematic viscosity that is the ratio of dynamic viscosity to density, μ/ρ. Additional definitions associated with the production and destruction terms are given as:[39]

$$\tilde{S} = \begin{cases} S + \hat{S} & \text{if } \hat{S} \geq -c_{v2}S \\ S + \frac{S(c_{v2}^2 + c_{v3}\hat{S})}{(c_{v3} - 2c_{v2})S - \hat{S}} & \text{if } \hat{S} < -c_{v2}S \end{cases} \tag{10}$$

$$S = \sqrt{\overrightarrow{\omega}\cdot\overrightarrow{\omega}}, \quad \hat{S} = \frac{\tilde{\nu}}{\kappa_t^2 d^2}f_{v2}, \quad f_{v1} = \frac{\chi^3}{\chi^3 + c_{v1}^3}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}, \quad f_{t2} = c_{t3}e^{-c_{t4}\chi^2} \tag{11}$$

and

$$\chi = \frac{\tilde{\nu}}{\nu}, \quad r = \min(\frac{\tilde{\nu}}{\tilde{S}\kappa_t^2 d^2}r_{\lim}), \quad g = r + c_{w2}(r^6 - r), \quad f_w = g\left(\frac{1 + c_{w3}^6}{g^6 + c_{w3}^6}\right)^{1/6} \tag{12}$$

where the vorticity vector is given by, $\overrightarrow{\omega} = \nabla \times \mathbf{u}$ and d represents the distance to the nearest wall.

The constants in the negative SA model are given as: $c_{b1} = 0.1355$, $\sigma = 2/3$, $c_{b2} = 0.622$, $c_{t3} = 1.2$, $c_{t4} = 0.5$, $\kappa_t = 0.41$, $c_{w1} = c_{b1}/\kappa_t^2 + (1 + c_{b2})/\sigma$, $c_{w2} = 0.3$, $c_{w3} = 2$ $c_{v1} = 7.1$, $c_{v2} = 0.7$ and $c_{v3} = 0.9$. κ and T denote the thermal conductivity and temperature, respectively, and are related to the total energy and velocity as,

$$\kappa T = \gamma(\frac{\mu}{P_r} + \frac{\mu_T}{P_{r_T}}) \left(E - \frac{1}{2}(u^2 + v^2 + w^2) \right) \tag{13}$$

where P_r and P_{r_T} are the Prandtl and turbulent Prandtl number that are set to 0.72 and 0.9, respectively. In the case of laminar flow, the governing equations reduce to the compressible Navier-Stokes equations, where the turbulence model equation is deactivated and the turbulence eddy viscosity, μ_T, in the fluid viscous stress tensor and the thermal conduction term vanishes.

For the purpose of the spatial discretization, the Cartesian viscous fluxes are rewritten in the following equivalent form:

$$\mathbf{F}_v^x = \mathbf{G}_{1j} \frac{\partial \mathbf{Q}}{\partial \mathbf{x}_j}, \quad \mathbf{F}_v^y = \mathbf{G}_{2j} \frac{\partial \mathbf{Q}}{\partial \mathbf{x}_j}, \quad \mathbf{F}_v^z = \mathbf{G}_{3j} \frac{\partial \mathbf{Q}}{\partial \mathbf{x}_j} \tag{14}$$

where the matrices $\mathbf{G}_{ij}(\mathbf{Q})$ are determined by $\mathbf{G}_{ij} = \partial \mathbf{F}_v^{x_i} / \partial(\partial \mathbf{Q}/\partial \mathbf{x}_j)$ for $i, j = 1, 2, 3$.

III. SUPG Stabilized Finite-Element Formulation

In the streamline upwind/Petrov-Galerkin method, the discretized system of equations is written as the following weighted residual formulation,

$$\begin{aligned}
&\int_\Omega \phi \left[\frac{\partial \mathbf{Q}_h}{\partial t} + \nabla \cdot (\mathbf{F}_e(\mathbf{Q}_h) - \mathbf{F}_v(\mathbf{Q}_h, \nabla \mathbf{Q}_h)) - \mathbf{S}(\mathbf{Q}_h, \nabla \mathbf{Q}_h) \right] d\Omega_k \\
&+ \sum_k \int_{\Omega_k} \left[\frac{\partial \phi}{\partial x_i} [\mathbf{A}_i] \right] [\tau] \left[\frac{\partial \mathbf{Q}_h}{\partial t} + \nabla \cdot (\mathbf{F}_e(\mathbf{Q}_h) - \mathbf{F}_v(\mathbf{Q}_h, \nabla \mathbf{Q}_h)) - \mathbf{S}(\mathbf{Q}_h, \nabla \mathbf{Q}_h) \right] d\Omega_k \\
&+ \mathcal{N}_\Gamma \left(\phi_j^+, \mathbf{Q}_b(\mathbf{Q}_h^+), \nabla_h \mathbf{Q}_h^+ \right) + \int_\Omega \left[\nu_s \nabla_h \phi \cdot \nabla_h \mathbf{Q} \right] d\Omega_k = 0
\end{aligned} \tag{15}$$

where ϕ is a continuous weighting function defined over the domain and $[\mathbf{A}_i]$ represents the inviscid flux Jacobians, defined as $[\mathbf{A}_1] = [\partial \mathbf{F}_e^x / \partial \mathbf{Q}_h]$, $[\mathbf{A}_2] = [\partial \mathbf{F}_e^y / \partial \mathbf{Q}_h]$, and $[\mathbf{A}_3] = [\partial \mathbf{F}_e^z / \partial \mathbf{Q}_h]$, respectively. The first integral in Eq. (15) corresponds to a standard Galerkin discretization, the second row provides dissipation for stability, and the contributions on the last row are penalty terms used to enforce boundary conditions and for capturing shocks. These terms will be discussed in greater detail in subsequent sections. The stabilization matrix, $[\tau]$, is used to compensate for lack of dissipation in the stream-wise direction[16] and for inviscid flows, is obtained using the following definitions,[42]

$$[\tau]^{-1} = \sum_{j=1}^{M} \left| \frac{\partial \phi_j}{\partial x_i} [\mathbf{A}_i] \right| \tag{16}$$

$$\left| \tfrac{\partial \phi_j}{\partial x_i} [\mathbf{A}_i] \right| = [\mathbf{T}] \, |\Lambda| \, [\mathbf{T}]^{-1} \tag{17}$$

Here, M corresponds to the number of basis functions within the element and the repeated index i implies summation over all the values of i ($i = 1, 2, 3$). ϕ_j denotes the polynomial basis function associated with each node and is the same as the weighting function. $[\mathbf{T}]$ and $[\Lambda]$ denote, respectively, the matrix of right eigenvectors and the diagonal matrix of eigenvalues of the left hand side matrix in Eq. (17). The inverse of the stabilization matrix is evaluated at each Gaussian quadrature point for volume integrations and $[\tau]$ is then obtained by means of local matrix inversion. To maintain design order-of-accuracy for viscous flows, additional terms are required when the Reynolds number is decreased and viscous terms dominate.[17] Specifically, the stabilization matrix in this case is appended with a viscous contribution, given as,

$$[\tau]^{-1} = \sum_{j=1}^{M} \left(\left| \frac{\partial \phi_j}{\partial x_i} [\mathbf{A}_i] \right| + \frac{\partial \phi_j}{\partial x_i} [\mathbf{G}_{ik}] \frac{\partial \phi_j}{\partial x_k} \right) \tag{18}$$

where the summation in the second term is taken over the repeated indices, i and k $(i,k=1,2,3)$, and the definitions of $[\mathbf{G}_{ik}]$ corresponds to those given in Eq. (14).

Integrating the volume integral on the first line of Eq. (15) by parts results in both volume and surface integral contributions so that the resulting system of equations to be discretized can be written as:

$$\begin{aligned}
&\int_{\Omega_k}\left[\phi\frac{\partial\mathbf{Q}_h}{\partial t}-\nabla\phi\cdot(\mathbf{F}_e(\mathbf{Q}_h)-\mathbf{F}_v(\mathbf{Q}_h,\nabla\mathbf{Q}_h))-\phi\mathbf{S}(\mathbf{Q}_h,\nabla\mathbf{Q}_h)\right]d\Omega_k\\
&+\sum_k\int_{\Omega_k}\left[\frac{\partial\phi}{\partial x_i}[\mathbf{A}_i]\right][\tau]\left[\frac{\partial\mathbf{U}_h}{\partial t}+\nabla\cdot(\mathbf{F}_e(\mathbf{Q}_h)-\mathbf{F}_v(\mathbf{Q}_h,\nabla\mathbf{Q}_h))-\mathbf{S}(\mathbf{Q}_h,\nabla\mathbf{Q}_h)\right]d\Omega_k\\
&+\int_{\partial\Omega_k\cap\partial\Omega}\phi\left(\mathbf{F}_e(\mathbf{Q}_b)-\mathbf{F}_v(\mathbf{Q}_b,\nabla\mathbf{Q}_h)\right)\cdot\mathbf{n}\;dS+\mathcal{N}_\Gamma\;(\phi_j^+,\mathbf{Q}_b(\mathbf{Q}_h^+),\nabla_h\mathbf{Q}_h^+)+\int_\Omega\left[\nu_s\nabla_h\phi\cdot\nabla_h\mathbf{Q}\right]d\Omega_k=0
\end{aligned}\tag{19}$$

Note that the last row now includes the same terms as in Eq. (15) in addition to the surface integral resulting from the integration by parts. The first and second terms on the last row are used in implementing the boundary conditions.

On solid walls, either strong or weak enforcement of the boundary conditions can be used. For strong enforcement, $\mathcal{N}_\Gamma$ is not used, while the velocity components are initialized to zero. To maintain no-slip velocity at the wall during the iterative process, the appropriate residual values are set to zero and the linearization matrix is modified so that unity on the diagonal is the only term surviving on the row. For the energy equation, a similar treatment is used when a constant temperature wall is desired, while the normal gradient of the temperature is set to zero for an adiabatic wall. By following this procedure with strong boundary conditions, the computation of viscous fluxes is unnecessary and the effects are automatically accounted for through the matrix.[1]

In the results presented in this paper, weak boundary conditions are used, which are adopted from techniques originally developed for discontinuous-Galerkin schemes,[43] and taken directly from Ref. 25. As reported in Ref. 44, and submitted to the 4th High-Order Workshop by the current authors, using these boundary conditions enabled super convergence with SUPG stabilized finite-elements for a laminar Joukowski airfoil case. Ref. 45 also demonstrated that weak boundary conditions, with appropriately defined cost functions, provide adjoint solutions that vary smoothly near solid walls. Although not used in this work, an alternate approach that is demonstrated to yield smooth adjoint solutions is that of Ref. 46.

As mentioned previously, $\mathcal{N}_\Gamma$ in Eq. (19) represents penalty terms, which are given as

$$\begin{aligned}
\mathcal{N}_\Gamma=&-\int_{\partial\Omega_k\cap\partial\Omega}(\mathbf{G}_{i1}(\mathbf{Q}_h)\frac{\partial\phi_j^+}{\partial\mathbf{x}_i},\mathbf{G}_{i2}(\mathbf{Q}_h)\frac{\partial\phi_j^+}{\partial\mathbf{x}_i},\mathbf{G}_{i3}(\mathbf{Q}_h)\frac{\partial\phi_j^+}{\partial\mathbf{x}_i})\cdot(\mathbf{Q}_h-\mathbf{Q}_b)\mathbf{n}\;dS\\
&+\int_{\partial\Omega_k\cap\partial\Omega}\eta_p\mathbf{G}(\mathbf{Q}_h)(\mathbf{Q}_h-\mathbf{Q}_b)\mathbf{n}\cdot\phi_j^+\mathbf{n}\;dS
\end{aligned}\tag{20}$$

In Eq. (20), ϕ_j^+ represents the basis functions associated with the element immediately adjacent to the boundary evaluated at the wall. The variable $\mathbf{Q}_b$ represents a boundary state that reflects the state quantities desired at the wall, and $\mathbf{Q}_h$ represents the dependent variables evaluated at the wall obtained from the adjacent element. The gradients are also evaluated from the element adjacent to the wall. The penalty parameter, η, is given by

$$\eta_p=\frac{(P+1)(P+D)(S_k^+)}{(2D)(V_k^+)}\tag{21}$$

where P is the order of the basis functions, D represents the space dimensions, and V_k^+ and S_k^+ represent the volume and surface area, respectively, of the element adjacent to the boundary.

In the far field, Roe's approximate Riemann solver[7] is used to evaluate the inviscid terms in the first integral on the last row of Eq. (19) and viscous stresses are assumed negligible. Because Dirichlet boundary conditions are not used on these boundaries, the penalty term, $\mathcal{N}_\Gamma$ is not required.

IV. Time Advancement

To advance the solution toward a steady state, the density, velocity components, temperature, and the turbulence-model working variable are tightly coupled and updated using a Newton-type algorithm. Although there are differences between the current approach and those used in Refs. 47 and 48, there are many similarities and elements are borrowed from both approaches.

Here, an initial update to the flow variables is computed using a locally varying time-step parameter that is multiplied by the current CFL number, which is adjusted during the iterative process to provide global convergence. Using the full update of the variables, the L_2 norm of the unsteady residual is compared to its value at the beginning of the iteration. If the L_2 norm after the update is less than one half of the original value, the steady residual is then computed and the CFL number is doubled if the steady residual does not increase by more than 20 percent; otherwise the CFL number is left unchanged. If the L_2 norm reduction target is not met, a line search is conducted to determine an appropriate relaxation factor. Before conducting the line search however, the solution after applying the full update is examined, and the maximum value for the line-search parameter is limited if either density or temperature is being driven negative. If the maximum value for the relaxation factor is less than a predetermined small value (currently 0.02), the update is rejected and the CFL number is reduced by a factor of ten. Otherwise, the L_2 norm of the residual is determined at four locations along the search direction between zero and the maximum update limit determined from realizability conditions described above. Using the four L_2 norm values of the residual, the optimal relaxation factor is determined by locating the minimum of a cubic polynomial curve fit through the samples. The solution is then updated using the relaxation factor and the CFL number is neither increased nor decreased.

Unlike in Ref. 47, during the process of determining whether to increase or decrease the time step, the result from the linear solver is not explicitly considered. At each nonlinear iteration, the linear system is approximately solved using the generalized minimal residual (GMRES) algorithm[49] with a preconditioner based on an incomplete lower upper (LU) decomposition with two levels of fill[50] and a Krylov subspace dimension of 200. Using GMRES, the residual norm for the linear system is guaranteed to be reduced with each additional search direction. Whether or not predetermined tolerances for the linear system are met, the updated flow variables are used during the line search. If the linear system is poorly solved, the nonlinear residual based on the flow equations can often still be reduced. In situations where this is not the case, the next update is inevitably rejected during the line search.

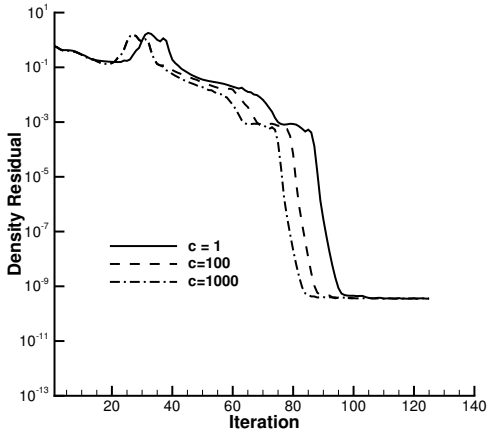
Note that the solution variables used for the finite-element discretization differ from those normally stored in the FUN3D finite-volume discretization, which solves for nondimensional conserved variables. The choice in solving for temperature directly in the finite-element discretization has been made to facilitate computations of real-gas flows where the equation of state is invariably given directly in terms of density and temperature. This choice is also made because when these variables are expressed using linear elements, their second derivatives are zero and, hence, the only contribution to the viscous terms on the second line of Eq. (19) are through variations in the viscosity.

Similar to Ref. 51, the turbulence working variable is expressed in terms of an alternate variable, η, which is scaled by a constant and replaces $\tilde{\nu}$ as the dependent variable,

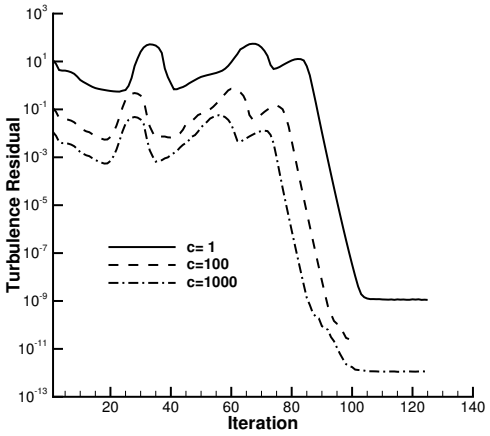
$$\tilde{\nu} = C\eta \tag{22}$$

Because the line search targets the reduction in the L_2 norm of all the variables, scaling to adjust the relative size of the turbulence-modeling residual with respect to the other equations may be beneficial. Essentially, the importance of the turbulence model can be either increased or decreased to effect the nonlinear convergence path. As with Ref. 51, and demonstrated in Fig. 3 for the ONERA M6 case presented later, using scale factors can have substantial effects on the convergence history. One should also note that with scaling, the initial level of the residual for the turbulence model is adjusted proportionally, with a corresponding adjustment in the achievable level of convergence.

While rescaling serves to rebalance the relative importance of the turbulence model in driving the convergence of the nonlinear system of equations, it also has an interesting effect on the linear system that does not correspond to simply scaling the entire row. To explain, consider first a block 6×6 entry in the global matrix taken from a row corresponding to an arbitrary point in a viscous flow simulation, and assuming a scale factor of 1000. Before scaling, as seen in Table 1, the last row has elements for the first 5 columns that are as much as 150 times larger than the sixth column, which is the linearization of the turbulence model with respect to $\tilde{\nu}$. Also observe that in the first five rows, the entries in the sixth column are small



(a) Density residual.



(b) Turbulence model residual.

Figure 3. Effect of scaling turbulence model on convergence: ONERA M6 wing on medium mesh. $M_\infty = 0.84$, $\alpha = 3.06^\circ$, $Re = 11.0 \times 10^6$.

relative to the other entries on their respective rows. The scaled variable, η, which is 1000 times less than $\tilde{\nu}$, replaces $\tilde{\nu}$ as the working variable for the turbulence model and thereby changes the entries in the block. In computing the last row in Table 2, the first five columns after scaling use the same linearization as before except that η assumes the role previously occupied by $\tilde{\nu}$. Because η is 1000 times smaller than $\tilde{\nu}$, the effect is that the magnitudes of these entries are reduced by a corresponding amount. However, the sixth column on the same row remains exactly the same as before the scaling. For similar reasons, the sixth column of the first five rows are correspondingly increased, but their relative magnitudes, even after scaling, remain small compared to the other entries on these rows. The effect of this procedure is that the coupling of the flow variables in the turbulence model equation is substantially weakened. This suggests that benefits may be attained for an algorithm that loosely couples the turbulence model and the flow equations by applying the scaling and updating the turbulence model prior to updating the flow equations, where the terms from the turbulence model appear only on the right-hand-side. This procedure may also be beneficial to iterative methods or preconditioners that do not include pivoting strategies.

Table 1. Example entries in residual-linearizations before scaling.

Residual	$\partial()/\partial\rho$	$\partial()/\partial u$	$\partial()/\partial v$	$\partial()/\partial w$	$\partial()/\partial T$	$\partial()/\partial\tilde{\nu}$
Density	0.6497E-01	0.3101E-01	-0.9070E+00	0.1031E+00	0.4628E-01	0.3098E-10
x-momentum	0.4524E-01	0.2656E-01	-0.2156E+00	0.2420E-01	0.3972E-01	-0.2309E-10
y-momentum	-0.6567E+00	0.3689E-01	0.8747E-01	0.1068E+00	-0.6489E+00	0.2256E-10
z-momentum	0.7791E-01	-0.4528E-02	0.1049E+00	0.1854E-01	0.7698E-01	-0.1963E-11
Energy	0.1591E+00	0.8348E-01	-0.2285E+01	0.2598E+00	0.1564E+00	0.6663E-10
Turbulence	0.1967E+00	0.3885E+00	-0.2744E+01	0.3233E+00	0.1403E+00	0.1778E-01

V. Blending Functions

During implementation, there are numerous nondifferentiable functions, such as $\max(x, y)$ and $\min(x, y)$, that could potentially impede convergence. For example, in the turbulence model, the result of a minimization comparison is used to limit the size of r in Eq. (12) to avoid floating-point overflow when it is subsequently raised to the sixth power. To provide similar, but differentiable, functionality, a smooth approximation is used for this term given by:

Table 2. Example entries in residual-linearizations after scaling.

Residual	$\partial()/\partial\rho$	$\partial()/\partial u$	$\partial()/\partial v$	$\partial()/\partial w$	$\partial()/\partial T$	$\partial()/\partial\eta$
Density	0.6497E-01	0.3101E-01	-0.9070E+00	0.1031E+00	0.4628E-01	0.3098E-07
x-momentum	0.4524E-01	0.2656E-01	-0.2156E+00	0.2420E-01	0.3972E-01	-0.2309E-07
y-momentum	-0.6567E+00	0.3689E-01	0.8747E-01	0.1068E+00	-0.6489E+00	0.2256E-07
z-momentum	0.7791E-01	-0.4528E-02	0.1049E+00	0.1854E-01	0.7698E-01	-0.1963E-08
Energy	0.1591E+00	0.8348E-01	-0.2285E+01	0.2598E+00	0.1564E+00	0.6663E-07
Turbulence	0.1967E-03	0.3885E-03	-0.2744E-02	0.3233E-03	0.1403E-03	0.1778E-01

$$\max(x,y) \approx \frac{1}{2}(x+y+\langle x-y\rangle) \tag{23}$$

Here, $\langle\phi\rangle$ designates a smoothed approximation to the absolute value and is given by

$$\langle\phi\rangle = \begin{cases} |\phi| & \text{if } |\phi| \geq \epsilon \\ 0.5(\frac{\phi^2}{\epsilon}+\epsilon) & \text{if } |\phi| < \epsilon \end{cases} \tag{24}$$

Although not discussed further, other functions, such as the Kreisselmeier-Steinhauser function[52] have also been used in place of Eq. (23) with good success. Note that in using these type of approximations, care needs to be given to the choice of ϵ to account for the relative scale of the variables being compared, and so that the results are independent of the physical scale of the mesh.

Another function that is used in multiple capacities is a smooth ramp that is zero until an initial threshold, x_s, is reached. The function then smoothly increases to unity when a terminating threshold, x_e , is achieved. Here, a simple trigonometric function is used and is given as

$$\psi(x:x_s,x_e) = \begin{cases} 0 & \text{if } x < x_s \\ \frac{1}{2}\left[\sin(\theta)+1\right] & \text{if } x_s \leq x \leq x_e \\ 1 & \text{if } x > x_e \end{cases} \tag{25}$$

where

$$\theta = \frac{\pi}{2}\left(\frac{2x-(x_s+x_e)}{x_e-x_s}\right)$$

VI. Shock Capturing

Without additional dissipation, the stabilized finite-element method would not contain suitable levels of dissipation for capturing strong gradients on meshes lacking proper resolution. One mechanism to achieve this goal is to augment the physical viscosity so that the width of the shock can be resolved by the local mesh size (see e.g., Refs. 53 and 54). The approach relies on the observation that the width of a shock h_s, the jump in velocity across the shock Δu, and the viscosity at the sonic point ν^*, are related by the following approximate expression,[55]

$$\frac{h_s \Delta u}{\nu^*} \approx 1 \tag{26}$$

The general procedure is to simply augment the local physical viscosity in the Navier-Stokes equations so that the thickness of the shock spans a single mesh cell, or perhaps a subcell for higher-order methods.[54] While straightforward implementation of technique has been examined in the course of the current work, a slight modification of this approach, that appears to be more robust in numerical experiments, is to simply add a penalty term to the weak formulation as given by the last term in Eq. (19). This is largely equivalent to adding extra viscosity to the diagonal elements of the $\mathbf{G}_{ij}(\mathbf{Q})$ terms in Eq. (14), but also adds viscosity to the continuity equation. In the current implementation, the velocity jump across shocks is approximated by simply using the local convective speed, and the desired shock width is specified to be the local cell

width, which is computed as the volume of the element divided by its surface area. Use of the simplified shock-capturing term instead of augmenting the physical viscosity guarantees that the additional dissipation is affected through a symmetric positive-definite matrix and numerical experiments have indicated that a somewhat more robust algorithm results.

Using the simplifying assumptions regarding the shock width and the velocity jumps, the form of the viscosity is given as

$$\nu = (\sqrt{u^2 + v^2 + w^2} + c) h_s \psi(\xi) \tag{27}$$

The function $\psi(\xi)$ is a switching function that attempts to smoothly activate the additional dissipation only in local regions as needed. For this, the blending function given by Eq. (25) is used with the lower bound of 0.05 determined through numerical experiments such that the switch is inactive for smooth flows. Similarly, an upper threshold of 0.1 has been determined as a conservative choice so that once the switch is initiated, it reaches a maximum value of unity relatively quickly. These values are used for all the computations shown in the present work. Without the capability to delay the presence of nonzero switch values, the switch would be nonzero throughout much of the field, thereby rendering the entire scheme to be only first-order accurate, independent of the polynomial degree of the basis functions.

The argument for the switching function is based on the shock-detection switch devised by Larsson,[56]

$$\xi = \begin{cases} 0 & \text{if } \nabla \cdot \vec{V} > 0 \\ \frac{-\nabla \cdot \vec{V}}{\max\left(1.5\omega, \frac{0.05c}{h_s}\right)} & \text{if } \nabla \cdot \vec{V} < 0 \end{cases} \tag{28}$$

Two modifications have been made to the Larsson switch to make it differentiable. First, the shock sensor is intended to only be used when the divergence of the velocity is negative, thereby activating only in compressive regions of the flow, however, a binary switch that tests the sign of the divergence of the velocity is not differentiable. Instead, the divergence is multiplied by a "squashing" function that maps the product of the divergence and the element length scale to a smooth function that is zero for positive values, and smoothly increases to unity over a small range of negative values, currently set to $(0.0, -0.001)$. Note that the multiplication with the length scale is required so the results do not depend on arbitrary scaling of the mesh that would result if using the divergence. To achieve this objective, a ramping function similar to that given in Eq. (25) is used.

A second modification concerns the determination of the maximum value of the two variables in the denominator, which are intended to turn off the shock switch inside boundary layers. Once again, to promote differentiability, the smooth maximum function given by Eq. (23) is used. As with the divergence, to avoid results that depend on scaling of the mesh, this function should be applied using ωh_s and c as arguments, and dividing by the length scale afterward.

VII. Results

VII.A. Manufactured Solutions

The method of manufactured solutions[57] is used to verify the correctness and establish the order-of-accuracy of the stabilized finite-element discretization as implemented in FUN3D. Because FUN3D does not currently support general partitioning for higher-order elements, the routines from FUN3D have been adapted to the testing environment provided by FUNSAFE,[20,25,35,36] which is a suite of finite-element codes originally developed by professors and research professors while at the Chattanooga campus of the University of Tennessee, and subsequently extended by numerous graduate students for further development in fluid dynamics,[45,58,59] electromagnetics,[60–62] and acoustic metamaterials.[63] Although the routines in FUN3D are newly developed, the essential data structures for both codes remains very similar, thereby facilitating incorporation of the FUN3D residual routines into the FUNSAFE infrastructure.

The forcing functions are derived from the following trigonometric functions:

$$\begin{aligned}
\rho &= \rho_0 \left[1 + \cos^2(\pi x)\cos^2(\pi y)\cos^2(\pi z)\right] \\
u &= u_0 \left[1 + \sin(\kappa\pi x)\cos(\kappa\pi x)\sin(\kappa\pi y)\cos(\kappa\pi y)\sin(\kappa\pi z)\cos(\kappa\pi z)\right] \\
v &= v_0 \left[1 + \cos^2(\kappa\pi x)\cos^2(\kappa\pi y)\cos^2(\kappa\pi z)\right] \\
w &= w_0 \left[1 + \sin^2(\kappa\pi x)\cos^2(\kappa\pi y)\sin^2(\kappa\pi z)\right] \\
T &= T_0 \left[1 + \sin^2(\kappa\pi x)\sin^2(\kappa\pi y)\sin^2(\kappa\pi z)\right] \\
\tilde{\nu} &= \tilde{\nu}_0 \left[\sin(\pi x)\cos(\pi x)\cos^2(\pi y)\sin(\pi z)\cos(\pi z)\right]
\end{aligned} \tag{29}$$

The function used for $\tilde{\nu}$ is specifically selected to ensure both positive and negative values to exercise the alternate flow paths within the negative SA turbulence model. For all element types, a series of sequentially refined meshes is used. For tetrahedral elements, individual meshes have been created, whereas the meshes for the other element types are generated from an initial hexahedral mesh, with random perturbations applied to the interior nodes to impose nonuniform spacing between the points. Note that for the pyramidal element, additional nodes are required to establish valid connectivity across the domain. The observed order-of-accuracy between two meshes may be evaluated as,

$$\text{order} = \frac{log(\frac{\epsilon_1}{\epsilon_2})}{log(\frac{h_1}{h_2})} \tag{30}$$

where $\epsilon_{1/2}$ represents the error on the coarse/fine mesh, respectively, computed using either the L_1 or L_2 norm. The mesh spacing is estimated as $h = (\frac{1}{N})^{\frac{1}{3}}$ for three-dimensional elements, where N is the total number of nodes in the field. Note that in the above evaluation for the observed order, the coefficient in the discretization error is assumed to be independent of the mesh size, which is true once in the asymptotic range.

For numerical integration, when the dependent variable is represented by a polynomial basis of order P, volume integrals are evaluated using quadrature formulas that exactly integrate polynomials of order $2P$. In some applications, customized quadrature rules or specialized formulas are sometimes used for certain element geometries. The use of these rules in some cases can reduce the number of sampling points required to obtain rank-sufficient matrices, or for avoiding numerical difficulties such as locking in structural elements. Use of these rules, and the effects on convergence, should be carefully investigated prior to implementation. To this end, customized quadrature rules are not used in the current work.

The achieved order-of-accuracy for linear (second-order) and quadratic (third-order) tetrahedral, hexahedral, pentahedral, and pyramidal elements is shown in tables 3 to 6. As seen, all elements achieve their design order-of-accuracy. Numerical integration for tetrahedral, pentahedral, and hexahedral elements use standard Gauss quadrature points and weights readily available in the literature. The basis functions for pyramidal elements are derivable from hexahedral elements and, as such, the quadrature rules for hexahedral elements are used for numerical integration. Standard $2 \times 2 \times 2$ and $3 \times 3 \times 3$ product rules are used for the linear and quadratic pyramidal elements, respectively.

Table 3. Achieved order-of-accuracy for tetrahedral elements.

		Linear		Quadratic	
Nodes in mesh	Q	L_1	L_2	L_1	L_2
2930/18676	ρ :	2.4123	2.4454	3.2241	3.2802
2930/18676	u:	2.3004	2.3095	3.1406	3.1765
2930/18676	v:	2.2918	2.2400	3.1407	3.1540
2930/18676	w:	2.2032	2.1754	3.1725	3.2200
2930/18676	T :	2.4478	2.4914	3.3036	3.3747
2930/18676	$\tilde{\nu}$:	2.1644	2.1005	3.1353	3.1269
18676/128610	ρ :	2.0471	2.0763	2.9493	2.9687
18676/128610	u:	2.1066	2.1170	2.9123	2.9226
18676/128610	v:	2.0829	2.0895	2.9712	2.9828
18676/128610	w:	2.0305	2.0414	2.9347	2.9420
18676/128610	T:	2.0196	2.0418	3.0342	3.0466
18676/128610	$\tilde{\nu}$:	2.1083	2.1032	2.9478	2.9683

Table 4. Achieved order-of-accuracy for hexahedral elements.

		Linear		Quadratic	
Nodes in mesh	Q	L_1	L_2	L_1	L_2
1000/8000	ρ :	2.3055	2.2705	3.1294	3.0977
1000/8000	u:	2.0528	2.0717	3.2113	3.2297
1000/8000	v:	2.0380	1.9711	3.1272	3.1251
1000/8000	w:	1.8539	1.8767	3.1455	3.1729
1000/8000	T:	2.3514	2.3254	3.2318	3.1443
1000/8000	$\tilde{\nu}$:	2.2098	2.1697	2.9806	2.9272
8000/64000	ρ :	2.1220	2.1284	3.0577	3.1022
8000/64000	u:	2.0825	2.0874	3.0974	3.0994
8000/64000	v:	2.0538	2.0358	2.9273	2.8960
8000/64000	w:	2.0011	2.0099	3.1026	3.1066
8000/64000	T:	2.1437	2.1350	3.0259	3.0281
8000/64000	$\tilde{\nu}$:	2.0822	2.0885	2.8774	2.8064

Table 5. Achieved order-of-accuracy for pentahedral elements.

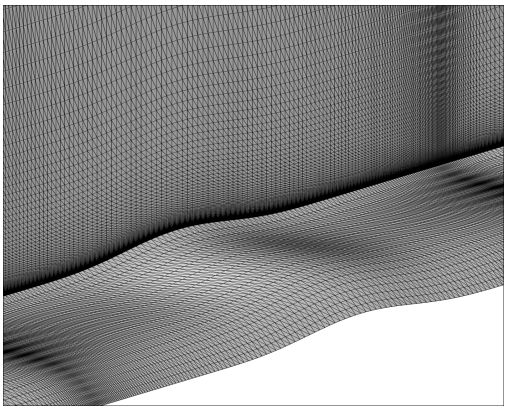
		Linear		Quadratic	
Nodes in mesh	Q	L_1	L_2	L_1	L_2
1000/8000	ρ :	2.3447	2.3068	2.9834	2.9667
1000/8000	u:	1.9678	1.9719	3.0190	3.0300
1000/8000	v:	1.9249	1.8475	2.8880	2.8945
1000/8000	w:	1.8937	1.8816	3.0392	3.0398
1000/8000	T:	2.3909	2.3346	2.9921	2.9579
1000/8000	$\tilde{\nu}$:	2.2234	2.1192	2.8688	2.8795
8000/64000	ρ :	2.1386	2.1355	2.9172	2.9410
8000/64000	u:	2.0759	2.0657	2.9496	2.9504
8000/64000	v:	2.0350	2.0017	2.8123	2.7465
8000/64000	w:	2.0394	2.0461	2.9759	2.9917
8000/64000	T:	2.1475	2.1381	2.9206	2.9298
8000/64000	$\tilde{\nu}$:	2.1137	2.0872	2.8358	2.8159

Table 6. Achieved order-of-accuracy for pyramidal elements.

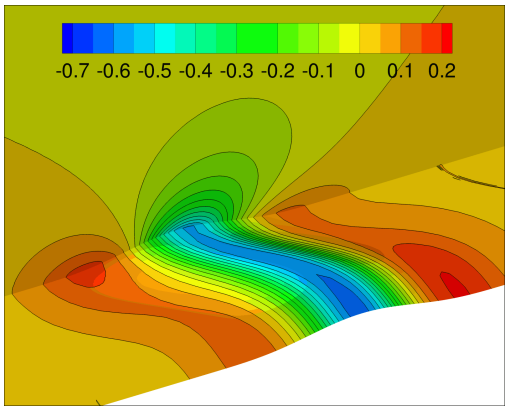
		Linear		Quadratic	
Nodes in mesh	Q	L_1	L_2	L_1	L_2
1729/14859	ρ :	2.1257	2.0924	3.0275	3.0515
1729/14859	u:	1.9718	1.9832	3.0217	3.0254
1729/14859	v:	1.9762	1.9127	2.9956	2.9860
1729/14859	w:	1.8902	1.9475	2.9379	2.9393
1729/14859	T:	2.1196	2.1193	3.0815	3.0644
1729/14859	$\tilde{\nu}$:	2.1055	2.0836	3.0200	2.9890
14859/123319	ρ :	2.0556	2.0561	3.0188	3.0388
14859/123319	u:	2.0148	2.0258	2.9853	2.9818
14859/123319	v:	2.0004	1.9926	2.9426	2.9112
14859/123319	w:	1.9809	1.9852	2.9732	2.9839
14859/123319	T:	2.0556	2.0570	3.0218	3.0089
14859/123319	$\tilde{\nu}$:	2.0421	2.0503	2.9749	2.9501

VII.B. 3D Swept Bump

To demonstrate the increased accuracy of the SUPG scheme over the finite-volume scheme on tetrahedral meshes, a simulation, initially reported in Ref. 28, is repeated and results are shown in Figs. 4, 5, and 6. Repeating the simulations here is for completeness to illustrate some important points regarding the relative accuracy of the SUPG scheme and the finite-volume scheme. The geometry, depicted in Fig. 4, is a swept three-dimensional bump in a channel, which is a verification case described on the NASA Turbulence Modeling Resource (TMR) web site.[64] A tetrahedral mesh, shown in Fig. 4(a) is used to illustrate the geometry, whereas nominal contours of pressure coefficient are shown in Fig. 4(b) to illustrate the problem. Profiles of the v-component of velocity, obtained from simulations with linear elements, are shown below in Fig. 5. A reference solution, obtained using the finite-volume scheme on a hexahedral mesh with 59 million nodes, is also included as a datum for comparison. Finite-element and finite-volume results obtained on tetrahedral meshes that have been derived from the hexahedral meshes are also shown. Note that division of the hexahedrons has been done so that all elements are cut in the same direction to introduce bias into the mesh because difficulties associated with uniform biasing of the elements are known to be somewhat problematic for the finite-volume scheme as reported in Refs. 28, 65 and 66. As seen in Fig. 5(a), the finite-volume solution on the tetrahedral mesh is quite poor, even with almost a million nodes in the mesh. As such, results on coarser meshes will not be shown as they simply degrade further. Instead, finite-volume solutions computed on hexahedral meshes are used to provide comparisons with the finite-element scheme on tetrahedral meshes. Velocity profiles obtained using the SUPG scheme on the tetrahedral mesh agree very well with the reference solution, as well as a finite-volume solution obtained on a pure hexahedral mesh. On coarser meshes, the finite-element scheme continues to achieve accuracy on tetrahedral meshes comparable to the finite-volume scheme on hexahedral meshes. Finally, note that the SUPG scheme achieves better accuracy on a tetrahedral mesh with only 18 thousand nodes, than is achieved by the finite-volume mesh with almost one million nodes. While the intentional bias introduced into the mesh clearly adversely effects the finite-volume scheme, the finite-element scheme is much more impervious to the biasing.



(a) Tetrahedral mesh for 3D swept bump.



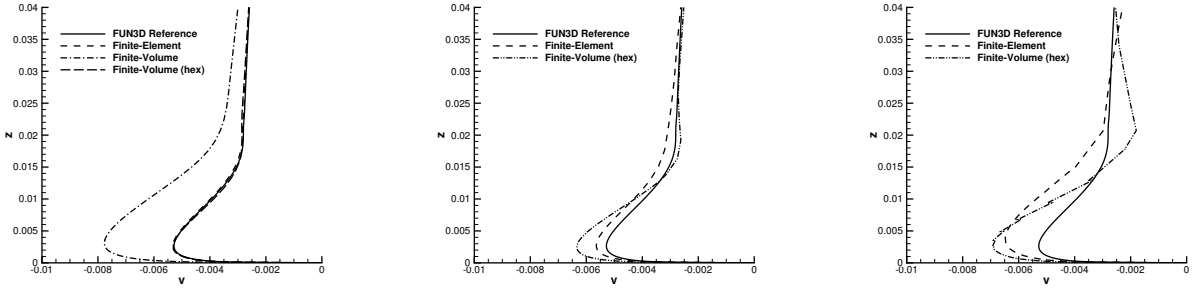
(b) Pressure contours for 3D swept bump.

Figure 4. Mesh and contours of pressure coefficient for 3D swept bump.

To assess potential gains in accuracy for the finite-element scheme that may be realized on hexahedral meshes, a comparison of results obtained on both tetrahedral and hexahedral meshes is shown in Fig. 6. On the finest mesh, no substantive improvement is observed by using hexahedrons when comparing the solutions against the reference solution, but, as seen in Fig. 6(b) and Fig. 6(c), benefits of using hexahedral meshes become more apparent as the grid is systematically coarsened. In summary, the significance of the cumulative results is that the accuracy of the stabilized finite-element scheme on tetrahedral meshes can be comparable to the finite-volume scheme on hexahedral meshes and that further gains in accuracy in the finite-element solutions can be achieved using hexahedral meshes.

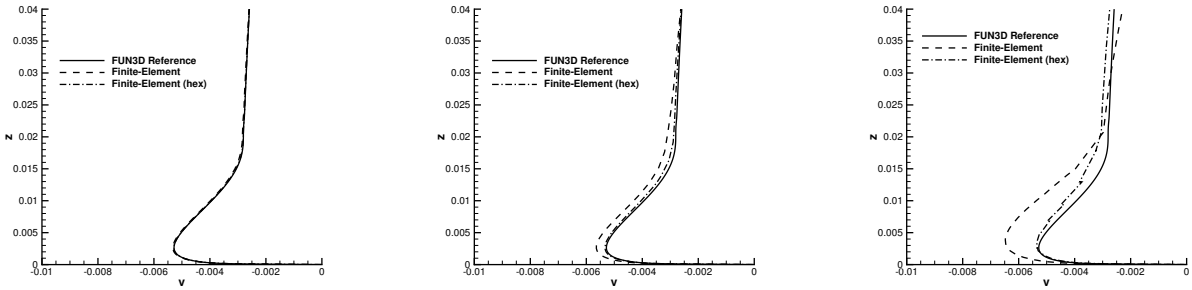
VII.C. Shock Capturing

To demonstrate the effectiveness of the shock capturing method, two examples are provided. The first example is a transonic flow over the wing used for the Third Drag Prediction Workshop.[67,68] This particular combination of geometry and flow conditions allows computations to be run beginning from freestream values



(a) 966 thousand node mesh. (b) 129 thousand node mesh. (c) 18 thousand node mesh.

Figure 5. Profiles of v-velocity.



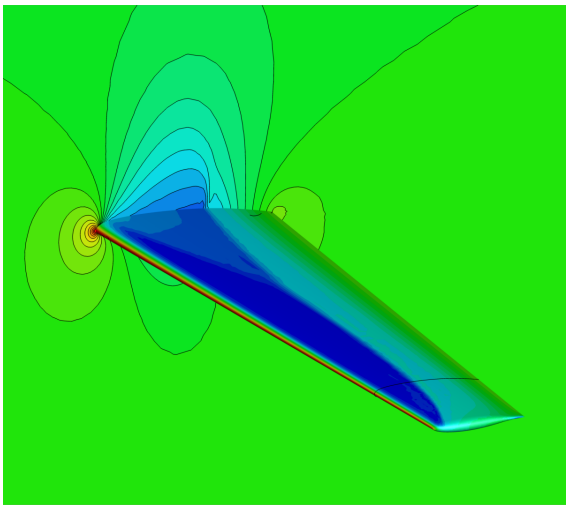
(a) 966 thousand node mesh. (b) 129 thousand node mesh. (c) 18 thousand node mesh.

Figure 6. Profiles of v-velocity obtained on hexahedral and tetrahedral elements.

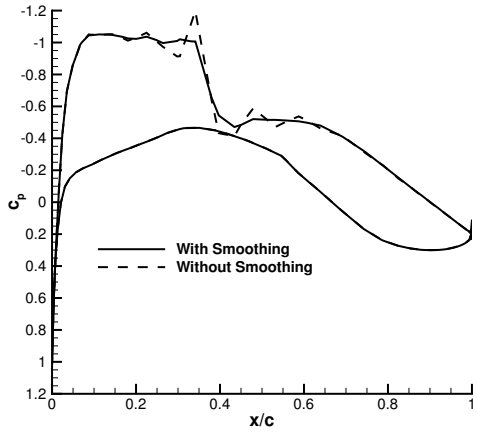
without the need for using the shock smoothing function, thereby allowing the effects of the shock capturing algorithm to be observed in isolation while still achieving iterative convergence. The wing, depicted in Fig. 7(a), is simulated at a freestream Mach number of 0.76, an angle-of-attack of 0.5°, and a Reynolds number, based on the mean aerodynamic chord, of $Re = 5.0 \times 10^6$. From Fig. 7(a) a shock is apparent on the upper surface of the wing, extending from the wing root outward to the tip. Computed pressure distributions have been obtained along the black line positioned toward the wing tip, and are shown in Fig. 7(b). Without the shock smoothing, large oscillations appear ahead of and behind the shock location. Using the shock sensor, however, these oscillations are essentially eliminated.

A second, more demanding example, is modeled after the circular cylinder case originally considered to illustrate the difficulties in computing hypersonic flows with finite-volume discretizations on tetrahedral meshes.[65,66,69,70] For the test case, a two-dimensional quadrilateral mesh is extruded into a hexahedral mesh with ten spanwise locations. A second mesh is then generated by cutting the hexahedral mesh into tetrahedrons, where the cutting is performed identically in each cell to introduce directional bias into the simulations. For this specific test, the initial two-dimensional mesh consists of 61 nodes distributed circumferentially around the cylinder, with 65 nodes extending from the surface of the cylinder to the outer boundary. Although any influence on the solution caused by the biasing will eventually diminish with grid convergence, the use of the coarse mesh is intended to exacerbate weaknesses with the discretization. A view of the cylinder on both the hexahedral mesh and the tetrahedral mesh is seen in Fig. 8. As expected, the tetrahedral mesh exhibits uniform biasing of the diagonal elements on the surface of the cylinder as seen in Fig. 8(b).

The flow conditions correspond to a freestream Mach number of 8.0, and a Reynolds number, based on the radius of the cylinder, of 300, 000. For all the simulations, a constant temperature wall is employed using the adiabatic wall temperature based on freestream values. Note that similar results have also been obtained for a Reynolds number of 3, 000, 000 with no change in conclusions. A baseline solution, which is considered the standard for comparison, is obtained on the hexahedral mesh by using the finite-volume scheme with the LDFSS[71] flux function and a van Albada flux limiter[72] augmented with a pressure limiter.[6] Solutions are



(a) Pressure contours with slice location.



(b) Pressure distributions.

Figure 7. Pressure distribution for transonic wing. $M_\infty = 0.76$, $\alpha = 0.5^\circ$, $Re = 5.0 \times 10^5$.

then attempted on the tetrahedral mesh and the results are compared with those from the hexahedral mesh.

To obtain the solution for the finite-volume scheme, 2000 iterations with first-order accuracy are first performed to allow an initial shock position to establish. Second-order accuracy is then initiated and the simulation is continued for 15,000 iterations, reducing the residual five orders-of-magnitude from its initial value. At several checkpoints, the upstream shock position and the skin friction values on the surface of the cylinder are periodically examined to verify that no substantive changes are occurring. Because of the limiter, convergence to machine zero is not a simple criteria to enforce, and freezing of this particular limiter is not permitted.

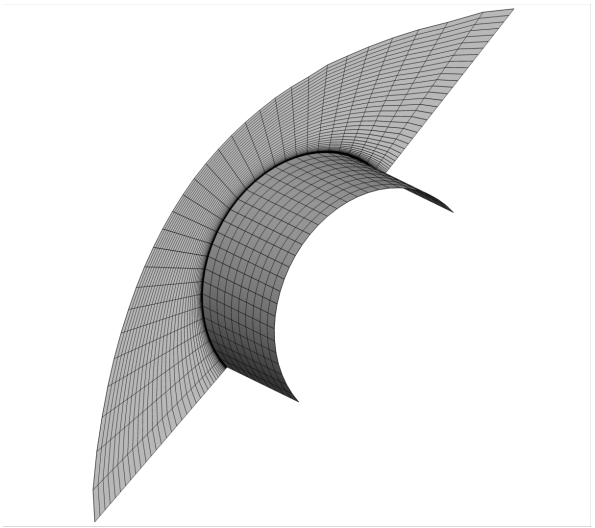
In running the stabilized finite-element scheme, two variants of the shock-smoothing algorithm are additively used. The first variation is a straight-forward use of the shock-smoothing algorithm as described above. The second variation is used only during initial transients to assist in establishing the final shock location. In this second variant, the switching function, $\psi(\xi)$, is simply set to a constant value of unity until the CFL number reaches 50, at which time it is gradually decreased based on the CFL number using the ramping function described by Eq. (25). Specifically, the switch is decreased to zero beginning at a CFL number of 50, and terminating at a CFL number of 500. At this point, the only additional dissipation added to the scheme is due to the shock sensor with the Larsson-based switch. The motivation for this strategy is based on the assumption that with the smoothing activated throughout the flow field, a shock, independent of where it appears, will be resolvable on the current mesh. Once the shock begins to establish its final position, this smoothing can then be reduced to zero.

The residual convergence history obtained with the SUPG scheme is shown in Fig. 9(a), with the corresponding history of the CFL number shown in Fig. 9(b). As seen, the CFL number for this computation is initialized somewhat arbitrarily to 0.1 where it remains over the first 20 iterations. At this point, the CFL number is seen to increase to slightly higher than 10 until approximately iteration 130, which is where the final shock position is established and the CFL rises very rapidly. The residual history is seen to exhibit a fairly "flat" shape until the shock position is established, at which point it rapidly drops to machine zero.

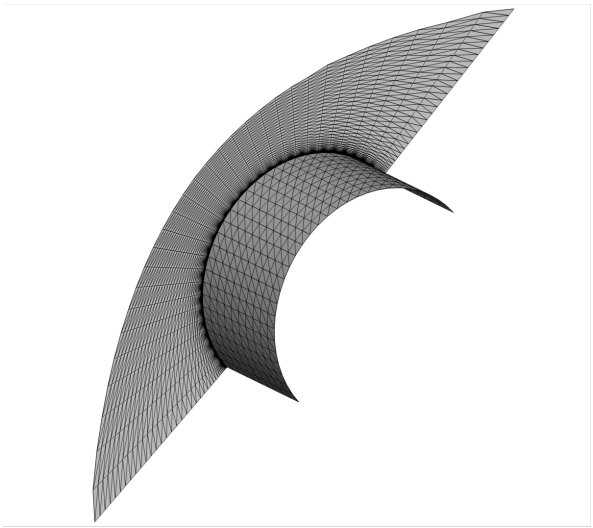
A side view of the pressure contours obtained using the finite-volume scheme on the hexahedral mesh is compared to the solution obtained with the finite-element scheme on the tetrahedral mesh in Fig. 10. As seen, both solutions are qualitatively similar, and neither one exhibits nonphysical behavior.

A quantitative comparison of the two solutions is provided in Fig. 11, which examines the pressure coefficient along a horizontal line extending from the leading edge of the cylinder to the outer boundary. As seen, the agreement between the finite-volume solution obtained on the hexahedral mesh, and the stabilized finite-element solution obtained on the tetrahedral mesh is quite good; the shock positions are in good agreement and neither solution exhibits overshoots.

In Refs. 65, 66, 69 and 70, accurate computation of the skin-friction values for the finite-volume scheme on tetrahedral meshes has been demonstrated to be an elusive goal. In Fig. 12, skin friction values com-



(a) Hexahedral mesh.



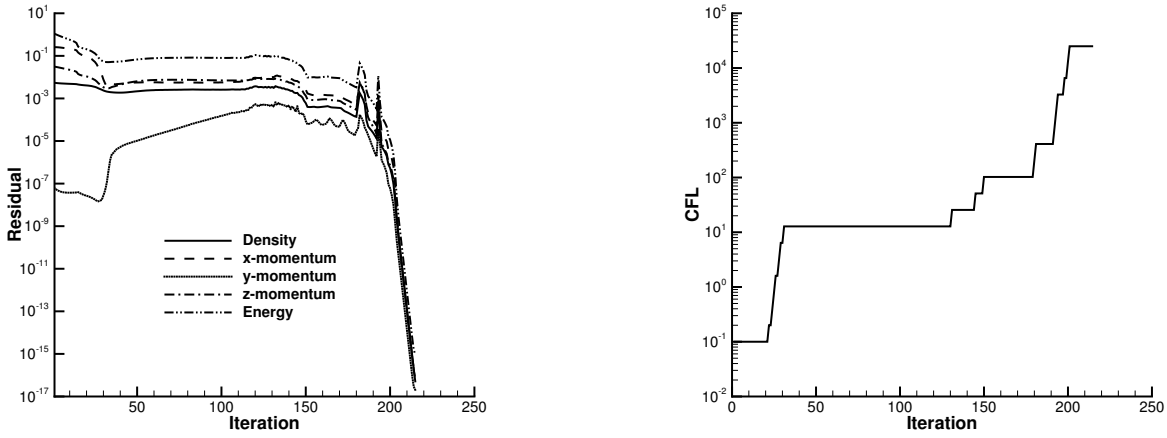
(b) Tetrahedral mesh.

Figure 8. Surface meshes for cylinder computations.

puted using the stabilized finite-element scheme on tetrahedrons are compared to those obtained using the finite-volume scheme on the pure hexahedral mesh. As seen, the finite-element solution has slightly higher maximum values and there is some modest bias in the solution, as evidenced by the slightly asymmetric values. Note that even on a symmetric geometry, some asymmetry will be expected unless the mesh is also symmetric; this is particularly true on coarser meshes.

In comparison, a solution on the tetrahedral mesh has also been attempted using the finite-volume scheme. Here, convergence criteria similar to that used for the solution on the hexahedral mesh could not be achieved using the LDFSS scheme. To achieve convergence, a flux-difference splitting scheme[7] is employed, with eigenvalue smoothing added to provide additional dissipation. Because of the coarseness of the mesh, the modified dissipation will manifest itself through the skin-friction values. To achieve a meaningful comparison between the finite-volume solution on both the hexahedral and tetrahedral meshes, the finite-volume solution on the hexahedral mesh has been repeated using the same flux function and limiter as used on the tetrahedral mesh. The results, depicted in Fig. 12(b), verify the trends observed in Refs. 65, 66, 69 and 70; namely, the results using the finite-volume scheme on the tetrahedral mesh exhibit strong variations across the span of the mesh.

A graphic summary of the results obtained using the finite-volume and finite-element schemes on tetrahedral meshes is presented in Fig. 13. Contours of skin friction computed using the finite-volume scheme on a hexahedral mesh are shown in Fig. 13(a) and, as expected, the solution is uniform across the span of the cylinder, reflective of the inherent symmetry in the mesh. In Fig. 13(b), the finite-element solution obtained on the biased tetrahedral mesh exhibits some slight spanwise variation, most notably immediately adjacent to the symmetry planes. Finally, the results for the finite-volume discretization on the tetrahedral mesh, shown in Fig. 13(c), are substantially poorer than the finite-element solutions on the same mesh and, as observable from the scale, yield higher skin-friction values than the finite-element solution. The stabilized finite-element solution is much less sensitive to the biased elements in the mesh and may provide a viable scheme for computing high-speed flows on tetrahedral meshes.



(a) Iterative convergence.

(b) CFL history.

Figure 9. Convergence history for circular cylinder. $M_\infty = 8.0$, $\alpha = 0.^\circ$, $Re = 3.0 \times 10^5$.

(a) Finite-volume hexahedral mesh.

(b) Finite-element tetrahedral mesh.

Figure 10. Pressure contours for supersonic cylinder. $M_\infty = 8.0$, $\alpha = 0.^\circ$, $Re = 3.0 \times 10^5$.

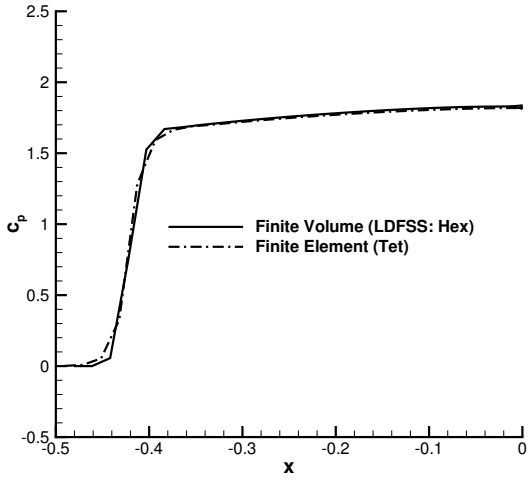
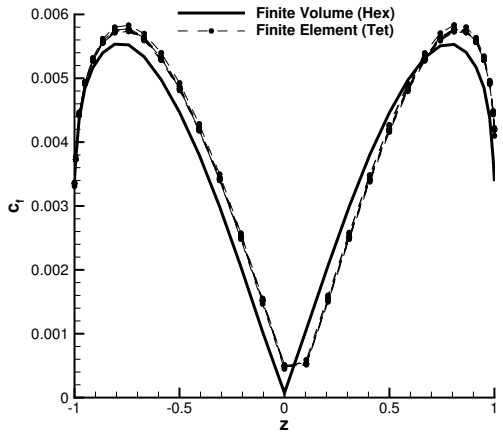
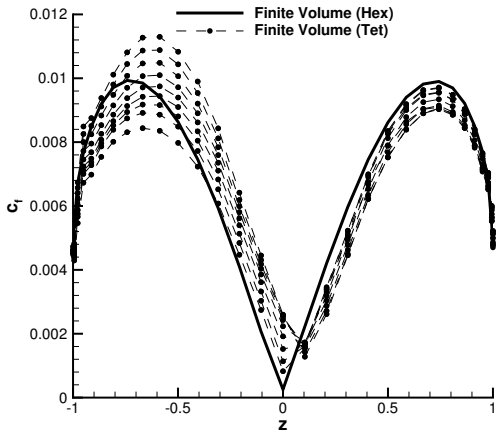


Figure 11. Pressure coefficient along stagnation streamline for circular cylinder. $M_\infty = 8.0$, $\alpha = 0.^\circ$, $Re = 3.0 \times 10^5$.

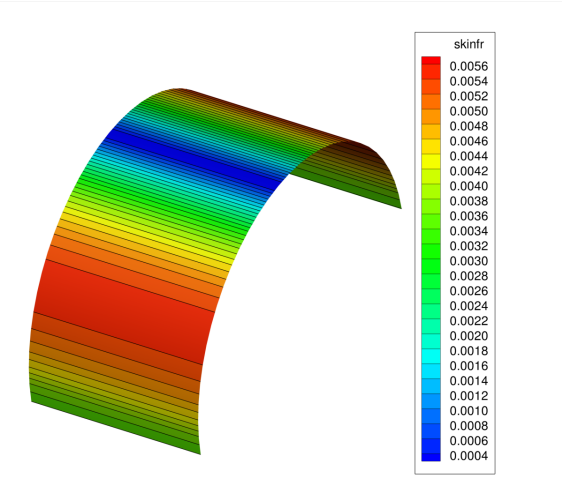


(a) Finite-volume scheme (LDFSS: hex) and finite-element scheme (tet).

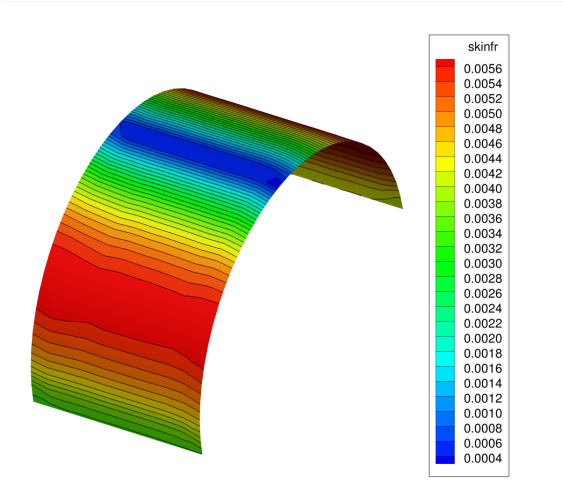


(b) Finite-volume scheme (FDS: hex) and finite volume scheme (FDS: tet).

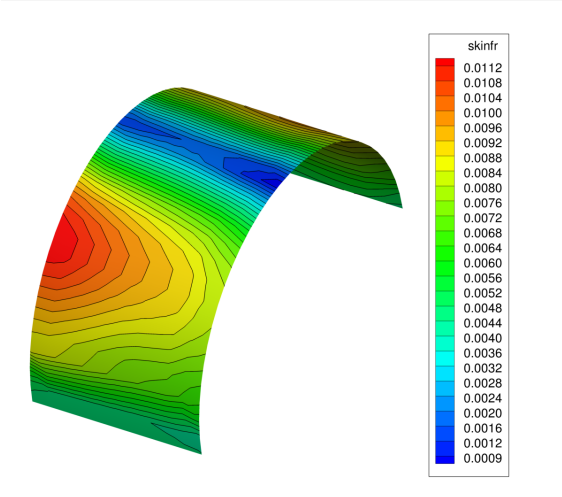
Figure 12. Computed skin friction for supersonic cylinder. $M_\infty = 8.0$, $\alpha = 0.^\circ$, $Re = 3.0 \times 10^5$.



(a) Finite-volume scheme (LDFSS: hex).



(b) Finite-element scheme (SUPG: tet).



(c) Finite-volume scheme (FDS: tet).

Figure 13. Skin friction contours for supersonic cylinder. $M_\infty = 8.0$, $\alpha = 0.^\circ$, $Re = 3.0 \times 10^5$.

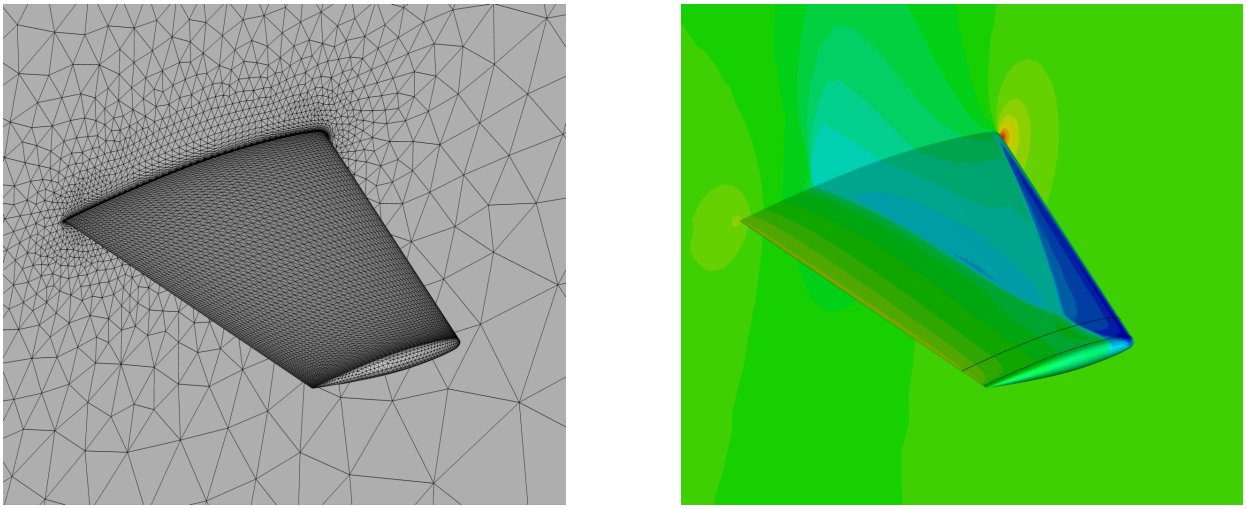
VII.D. Iterative Convergence and Timing Comparisons

To examine the effectiveness of the time-advancement method, simulations for transonic flow over the ONERA M6 wing[73] are initially considered. The flow conditions for the simulations correspond to a freestream Mach number of 0.84, an angle-of-attack of 3.06°, and a Reynolds number based on mean aerodynamic chord of 11.8×10^6. A sequence of four meshes is used for the simulations and comparisons in pressure distribution, residual convergence, and time to solution are made between the finite-element methodology and the baseline finite-volume algorithm. The parameters associated with each mesh are given in table 7. Note that the geometry is specified in millimeters so the Reynolds number used for the computation reflects scaling by the mean aerodynamic chord of 646.07 and the wall spacings correspond to estimated y^+ values of approximately 3, 2, 1.5, and 1.0, respectively. For comparison purposes, the finite-element and finite-volume schemes are both executed using the same number of cores, which is also indicated in the table.

Table 7. Mesh parameters for ONERA M6 calculations.

Mesh	Nodes	Tetrahedrons	Wall spacing	CPU Cores
Tiny	231,194	1,307,388	0.00360	16
Coarse	711,820	4,193,397	0.00240	48
Medium	2,307,525	13,667,813	0.00160	128
Fine	7,856,265	46,730,385	0.00107	320

The surface mesh on the smallest grid is shown in Fig. 14(a), with computed pressure contours shown in Fig. 14(b). Although the volume mesh is relatively coarse, the surface mesh has substantial resolution, which allows for well resolved shock structures, even on such a coarse mesh.



(a) Surface mesh for tiny grid.

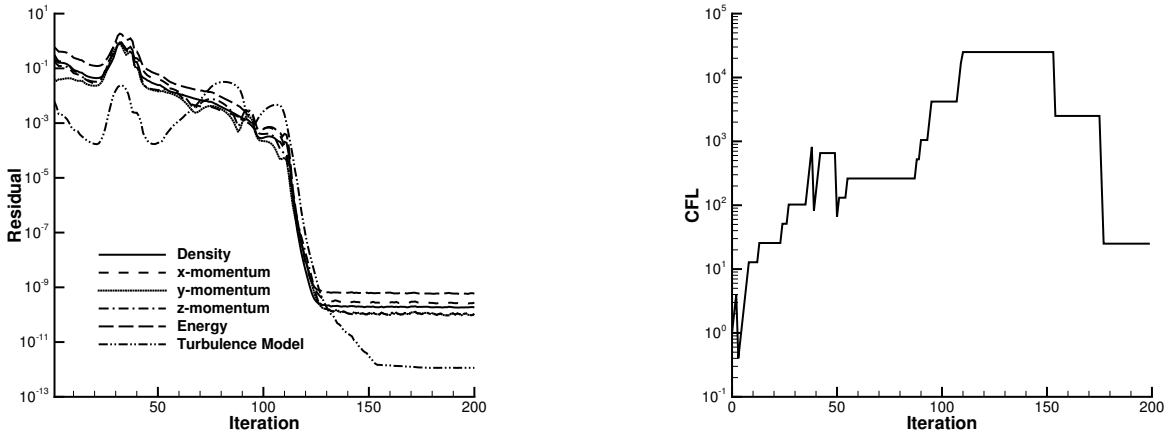
(b) Pressure contours on surface.

Figure 14. Surface mesh and pressure contours: ONERA M6 wing. $M_\infty = 0.84$, $\alpha = 3.06°$, $Re = 11.0 \times 10^6$.

Residual convergence for the finite-element scheme on the finest mesh is depicted in Fig. 15(a) with the corresponding CFL history shown in Fig. 15(b). As seen in Fig. 15(a), modest reductions in the residual are achieved over the the first 110 iterations, at which point, the domain of attraction to the root is obtained and drastic reductions in the residual are observed. The relatively small reductions in the residual prior to iteration 110 are due to the establishment of the shock structure in the flow field. As with the example depicted in Fig. 3, the residual for the turbulence model reaches a lower value than for the flow equations due to the scaling.

The CFL number for these simulations, as shown in Fig. 15(b), is initialized to unity and after a brief initial increase, is decreased to below one within the first 5 iterations. From there, it is systematically, although slightly erratically, increased to the maximum value allowed at which point the residual is rapidly reduced. After reaching machine zero, further reduction of the residuals can no longer be achieved and the line search, being unsuccessful, consequently reduces the CFL number. Note that while a maximum CFL

number of only 25,000 is used, this choice is somewhat arbitrary and values as high as 1 million have been used during testing. The current choice is only because the increased rate of convergence attributable to the higher CFL number has been observed to be fairly minor once the solution is close to the root, saving only a few iterations.



(a) Iterative convergence.

(b) CFL history.

Figure 15. Convergence history for ONERA M6 wing on finest mesh. $M_\infty = 0.84$, $\alpha = 3.06^\circ$, $Re = 11.8 \times 10^6$.

Pressure distributions along two spanwise locations are provided for comparison in Fig. 16. The spanwise positions are both toward the wing tip and correspond to the two black lines extending along the chord in Fig. 14. Note that while one of the lines is easily discernible, the other is located where the wing and the end cap join, and is somewhat difficult to visualize. Figure 16 demonstrates that there is very little variation in the pressure distributions observable in the finite-element solutions as the mesh is refined. Of particular interest, depicted in the close-up view in Fig. 16(e), is that the shock at the outer most spanwise station is present on all meshes except the tiny one. As seen in Fig. 16(f), somewhat more variation is observed in the finite-volume solutions where, at the outermost span-wise station, the simulation fails to resolve the shock wave even on the finest mesh. Note also that in the finite-volume simulation, no additional smoothing or flux limiting is used so that any dissipation is either physical dissipation or is inherent in the scheme. These results are consistent with those presented for the transonic airfoil in the introduction, although somewhat less dramatic because here, even the tiny mesh has relatively high surface resolution.

The iterative convergence between the baseline finite-volume scheme and the time-advancement scheme used for the finite-element algorithm is compared in Fig. 17. For convenience, the convergence of the finite-element scheme, previously presented in Fig. 15, is repeated here to facilitate comparison. The finite-element solver converges to machine precision in approximately 125 iterations, whereas the finite-volume scheme requires 20,000. Note that while both schemes use the negative SA model, for this case the finite-volume scheme uses first-order accurate convection because the residual could not otherwise be decreased to machine precision. As expected, the finite-element scheme converges in much fewer iterations than the finite-volume scheme simply because the algorithm makes far fewer assumptions in the linearization of the residual and uses a much stronger linear solver. Although not shown, only minor differences are observed in the pressure distributions obtained using the first-order and second-order accurate convective terms for the turbulence model.

In examining the convergence history, the majority of iterations required to converge the SUPG scheme occur in the first 110 steps, which is when the shock position is being established. To examine the convergence for a case without a shock, a subsonic case is considered and is presented in Fig. 18. As expected, the residual is reduced much more rapidly in this case, with the flow variables reaching machine precision in only 60 iterations, with the turbulence model requiring approximately 80 time steps. In contrast, a notable decrease in convergence rate is observed for the finite-volume scheme when compared to the convergence for the transonic case. Recall that in the transonic case, the finite-volume scheme was run using the negative SA model with first-order accurate convective discretization of the turbulence model because the residual would otherwise "hang". In the subsonic case, the negative SA model is again used, but second-order discretization

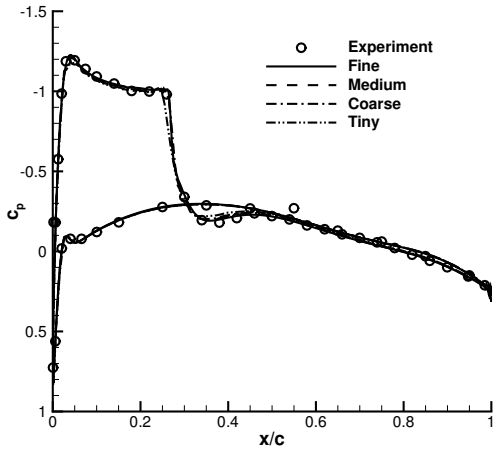
of the convective term in the turbulence model is used to render a more neutral comparison. The presence of the second-order differencing may likely be the cause in the difference in convergence between the transonic and subsonic cases for the finite-volume discretization.

While iterative convergence is interesting, the computer time to reach convergence is the metric that should ultimately be evaluated. The transonic and subsonic flows over the ONERA M6 wing are used for this initial assessment, simply because both the finite-element and finite-volume methodologies converge well and provide reasonably accurate results. However, there are several caveats that should be considered. First, in comparing the timings, both schemes are run on the same machine using the same number of processors. There have been no studies conducted with the finite-element scheme to determine optimal balancing of the number of nodes across processors to achieve good performance. Similarly, there has been no attempt to date to address coding or algorithmic issues with the SUPG scheme; the results are still preliminary and many improvements can be easily identified. Second, in comparing timings, the most meaningful measure would account for the accuracy of the obtained solution. In this vein, it would be interesting to compare results for the 3D swept bump presented earlier. There, the finite-element solution on a tetrahedral mesh with only 18 thousand mesh points is far superior to the finite-volume solution with almost 1 million nodes. Clearly, timing comparisons should be used as a general gauge and can not be universally applied based on a limited number of test cases. The purpose of the comparisons at this stage of development is to evaluate the relative cost to guide future development.

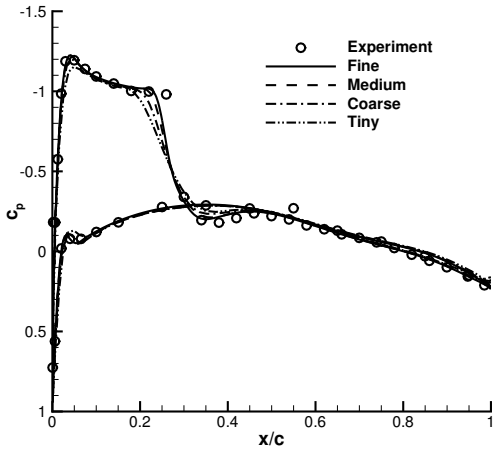
A summary of time to achieve iterative convergence for the transonic and subsonic ONERA M6 cases is provided in Fig. 19. In Fig. 19(a), the baseline finite-volume scheme appears to require approximately half the computer time as the finite-element scheme to converge the flow variables, while a somewhat smaller ratio is required to converge the turbulence model. Recall, however, that the finite-volume scheme uses only first-order accuracy for the convective terms in the turbulence model, which undoubtedly aids in convergence, especially since the scheme would not converge at all using second-order discretization for this term. In the subsonic case, the SUPG scheme clearly converges in less time than the finite-volume scheme, which has not reached machine precision after twice as much computer time.

To date, no consideration has been given to the performance of the finite-element code. The goal has been to simply evaluate the viability of the scheme and to demonstrate that much improved accuracy can be achieved when compared to the finite-volume scheme. Specifically, no effort has been devoted to cache considerations or culling wasteful and duplicative regions of code. A brief analysis of the run time indicates that the vast majority of the time is spent computing the linearization matrix, followed closely by the incomplete LU decomposition used as a preconditioner for GMRES. These operations are currently repeated at every iteration of the flow solver and clearly consume most of the computer time. Another important observation is that in previous studies for time-dependent flows,[25,36] the linearization can be frozen for as many as 50 time steps, thereby minimizing the computation required to recompute the matrix and perform an approximate LU decomposition.

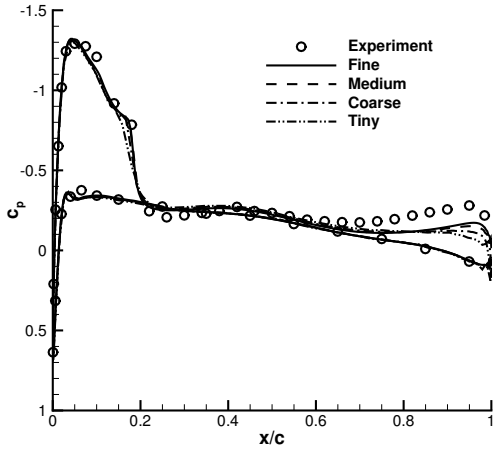
A final result is presented in Fig. 20. In examining the pressure distributions in Fig. 16, one could reasonably claim that the SUPG scheme achieves similar accuracy on either the medium- or coarse-sized mesh that is only achieved using the finite-volume scheme on the fine mesh. Comparing the time to solution for the SUPG scheme on the medium mesh with that of the finite-volume scheme on the fine mesh, the wall-clock time is observed to be approximately equal for the turbulence model to reach convergence. However, recall from Table 7 that the results on the fine mesh are obtained using 320 cores, whereas the results on the medium mesh are obtained using only 128 cores. Because the figure compares wall time and not total CPU time, the SUPG scheme ultimately requires less than half the total resources as the finite-volume scheme for similar accuracy.



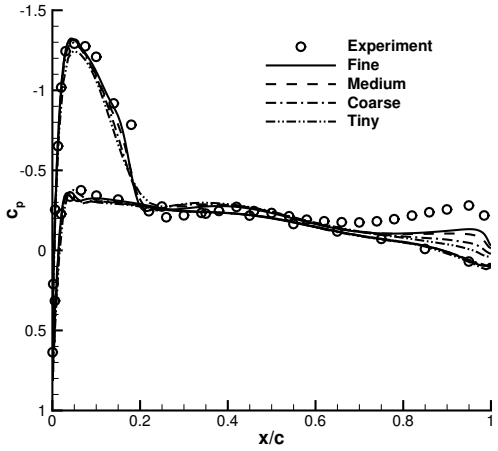
(a) Finite-element solution $\eta = 0.90$.



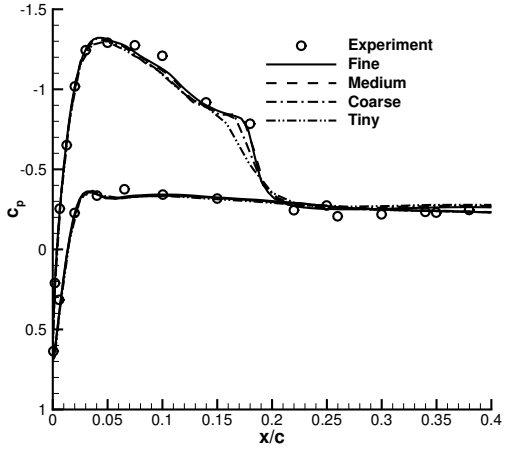
(b) Finite-volume solution $\eta = 0.90$.



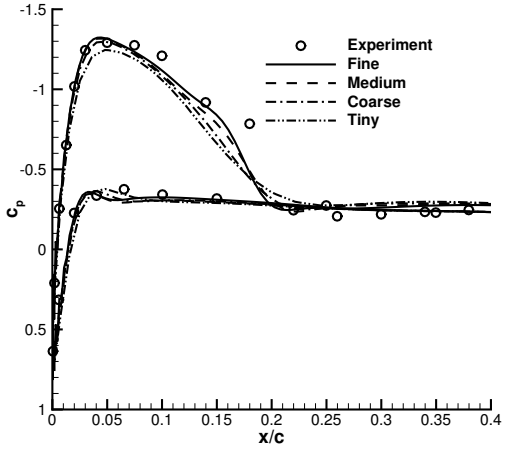
(c) Finite-element solution $\eta = 0.99$.



(d) Finite-volume solution $\eta = 0.99$.

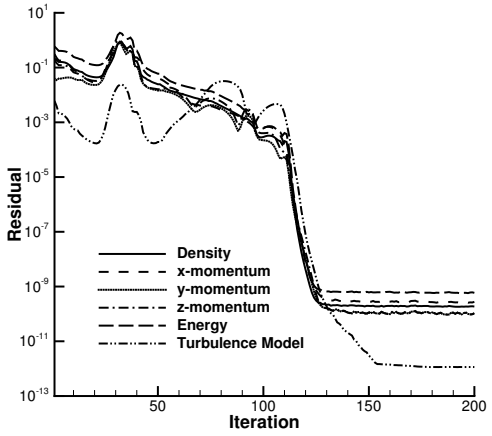


(e) Closeup view of finite-element solution $\eta = 0.99$.

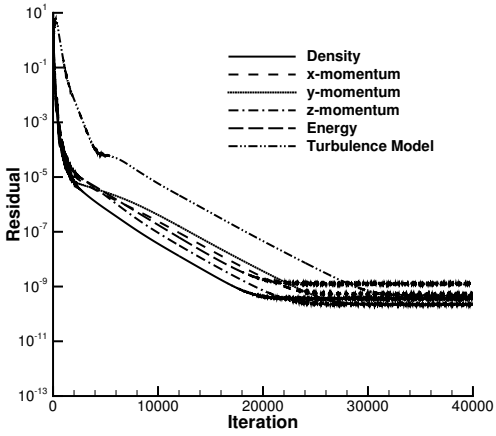


(f) Closeup view of finite-volume solution $\eta = 0.99$.

Figure 16. Pressure distributions for ONERA M6 wing. $M_\infty = 0.84$, $\alpha = 3.06^\circ$, $Re = 11.0 \times 10^6$.

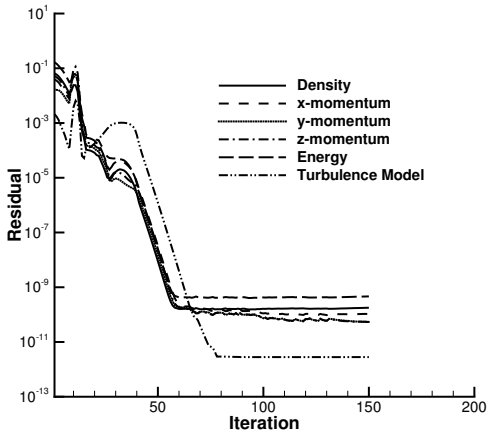


(a) Finite-Element Scheme.

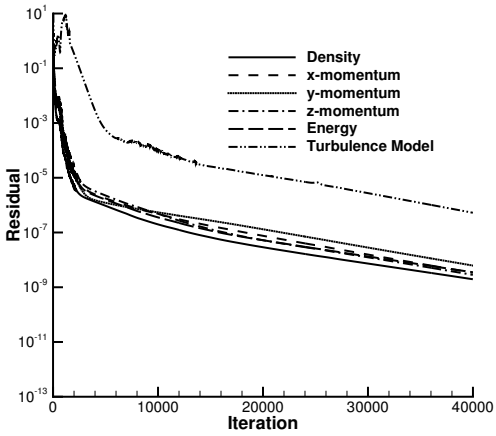


(b) Finite-Volume Scheme.

Figure 17. Convergence histories for ONERA M6 wing. $M_\infty = 0.84$, $\alpha = 3.06^\circ$, $Re = 11.0 \times 10^6$.

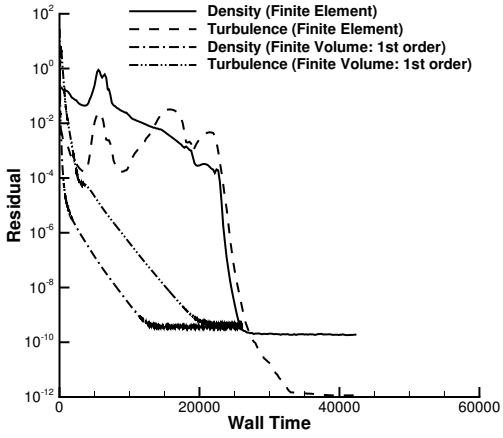


(a) Finite-Element Scheme.

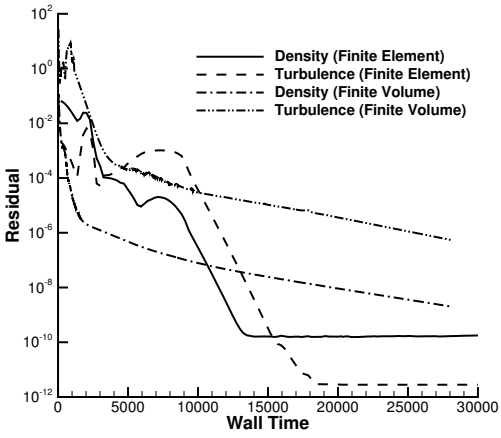


(b) Finite-Volume Scheme.

Figure 18. Finite-element and finite-volume convergence histories for ONERA M6 wing. $M_\infty = 0.84$, $\alpha = 3.06^\circ$, $Re = 11.0 \times 10^6$.



(a) Transcionic.



(b) Subsonic.

Figure 19. Convergence history for transonic and subsonic flow: ONERA M6 wing. $M_\infty = 0.84$, $\alpha = 3.06^\circ$, $Re = 11.0 \times 10^6$.

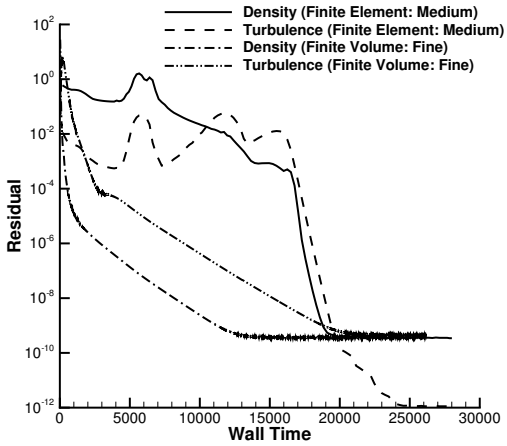


Figure 20. Finite-element and finite-volume convergence histories for ONERA M6 wing. $M_\infty = 0.84$, $\alpha = 3.06^\circ$, $Re = 11.0 \times 10^6$.

VIII. Summary and Future Work

A stabilized finite-element discretization has been developed and implemented as a linkable library within the FUN3D flow solver. The accuracy of the scheme has been verified for linear and quadratic basis functions on tetrahedral, hexahedral, prismatic, and pyramidal elements. A time-advancement algorithm has been developed and demonstrated for transonic and subsonic flows, as well as for a hypersonic test case. This hypersonic test case also serves as an initial evaluation of the mechanism used for capturing strong shocks. Through the test cases presented, the SUPG finite-element discretization is demonstrated to offer substantially improved accuracy over the finite-volume scheme on tetrahedral elements. Timing assessments indicate that while there has been no attempt to enhance performance, the SUPG finite-element scheme is competitive with the finite-volume scheme. Furthermore, when accuracy is also considered, the finite-element scheme could potentially offer significant reductions in the time required to reach a solution of given accuracy.

To date, no consideration has been given to the performance of the finite-element code; the goal has been to simply evaluate the viability of the scheme and to demonstrate that improved accuracy can be achieved when compared to the finite-volume scheme. A brief analysis of the run time indicates that the vast majority of the time is spent computing the linearization matrix, followed closely by the incomplete LU decomposition used as a preconditioner for GMRES. These operations are currently repeated at every iteration of the flow solver and clearly consume most of the computer time. These issues will be addressed during a rewrite of the code base into C++, which is currently under way.

FUN3D is currently undergoing major refactoring effort to better share common software components across multiple discretization options being incorporated into the code base. During this time, partitioning of the mesh for high-order discretization is obviously a priority, as is the development of h-p adaptive technology.

References

[1]Anderson, W. K. and Bonhaus, D. L., "An Implicit Upwind Algorithm for Computing Turbulent Flows on Unstructured Grids," *Computers and Fluids*, Vol. 23, No. 1, 1994, pp. 1–22.

[2]Anderson, W. K., Rausch, R. D., and Bonhaus, D. L., "Implicit/Multigrid Algorithm for Incompressible Turbulent Flows on Unstructured Grids," *Journal of Computational Physics*, Vol. 128, No. 2, 1996, pp. 391–408.

[3]Anderson, W. K. and Bonhaus, D. L., "Airfoil Design on Unstructured Grids for Turbulent Flows," *AIAA Journal*, Vol. 37, No. 2, 1999, pp. 185–191.

[4]Nielsen, E. J. and Anderson, W. K., "Aerodynamic Design Optimization on Unstructured Meshes Using the Navier-Stokes Equations," *AIAA journal*, Vol. 37, No. 11, 1999, pp. 1411–1419.

[5]Nielsen, E. J. and Anderson, W. K., "Recent Improvements in Aerodynamic Design Optimization on Unstructured Meshes," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1155–1163.

[6]Biedron, R. T., Carlson, J.-R., Derlaga, J. M., Gnoffo, P. A., Hammond, D. P., Jones, W. T., Kleb, B., Lee-Rausch, E. M., Nielsen, E. J., Park, M. A., Rumsey, C. L., Thomas, J. L., and Wood, W. A., "FUN3D Manual: 13.0," NASA TM-2016-219330, Langley Research Center, Aug. 2016.

[7]Roe, P. L., "Approximate Riemann Solvers, Parameter Vectors, and Difference Schemes," *Journal of Computational Physics*, Vol. 43, No. 2, Oct. 1981, pp. 357–372.

[8]Wang, Z. J., Fidkowski, K., Abgrall, R., Bassi, F., Caraeni, D., Cary, A., Deconinck, H., Hartmann, R., Hillewaert, K., Huynh, H. T., Kroll, N., May, G., Persson, P.-O., van Leer, B., and Visbal, M., "High-Order CFD Methods: Current Status and Perspective," *International Journal for Numerical Methods in Fluids*, Vol. 72, No. 8, July 2013, pp. 811–845.

[9]Hartmann, R. and Houston, P., "Adaptive Discontinuous Galerkin Finite Element Methods for the Compressible Euler Equations," *Journal of Computational Physics*, Vol. 183, No. 2, 2002, pp. 508–532.

[10]Fidkowski, K. J. and Darmofal, D. L., "Development of a Higher-Order Solver For Aerodynamic Applications," AIAA Paper 2004–436, 2004.

[11]Ceze, M. and Fidkowski, K. J., "Pseudo-transient Continuation, Solution Update Methods, and CFL Strategies for DG Discretizations of the RANS-SA Equations," AIAA Paper 2013–2686, 2013.

[12]Burgess, N. K., *An Adaptive Discontinuous Galerkin Solver for Aerodynamic Flows*, Ph.D. thesis, University of Wyoming, 2011.

[13]Burgess, N. K. and Mavriplis, D. J., "Robust Computation of Turbulent Flows Using a Discontinuous-Galerkin Method," AIAA Paper 2012–457, 2012.

[14]Oliver, T. A. and Darmofal, D. L., "An Unsteady Adaptation Algorithm for Discontinuous-Galerkin Discretizations of the RANS Equations," AIAA Paper 2007–3940, 2007.

[15]Oliver, T. A., *A High-Order, Adaptive, Discontinuous Galerkin Finite Element Method for the Reynolds-Averaged Navier-Stokes Equations*, Ph.D. thesis, Massachusetts Institute of Technology, 2008.

[16]Brooks, A. N. and Hughes, T. J. R., "Streamline Upwind/Petrov-Galerkin Formulation for Convection Dominated Flows with Particular Emphasis on Incompressible Navier-Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 32, No. 1–3, Sept. 1982, pp. 199–259.

[17]Shakib, F., Hughes, T. J. R., and Johan, Z., "A New Finite-Element Formulation for Computational Fluid Dynamics: X. The Compressible Euler and Navier-Stokes Equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 89, No. 1–3, Aug. 1991, pp. 141–219.

[18]Hughes, T. J. R., Franca, L. P., and Hulbert, G. M., "A New Finite-Element Formulation for Computational Fluid Dynamics: VIII. The Galerkin Least Squares Method for Advective-Diffusion Equations," *Computer Methods in Applied Mechanics and Engineering*, Vol. 73, No. 2, May 1989, pp. 173–189.

[19]Bazilevs, Y. and Akkerman, I., "Large Eddy Simulation of Turbulent Taylor–Couette Flow Using Isogeometric Analysis and the Residual-Based Variational Multiscale Method," *Journal of Computational Physics*, Vol. 229, No. 9, May 2010, pp. 3402–3414.

[20]Anderson, W. K., Wang, L., Kapadia, S., Tanis, C., and Hilbert, B., "Petrov–Galerkin and Discontinuous-Galerkin Methods for Time-Domain and Frequency-Domain Electromagnetic Simulations," *Journal of Computational Physics*, Vol. 230, No. 23, 2011, pp. 8360–8385.

[21]Anderson, K. and Wang, L., "A Perspective on High-Order Accurate Solvers for Field Equations," JRV Symposium; Four Decades of CFD: Looking Back and Looking Forward, 2013.

[22]Wang, L., Anderson, K., Erwin, T., and Kapadia, S., "Solutions of High-Order Methods for Three-Dimensional Compressible Viscous Flows," AIAA Paper 2012–2836, 2012.

[23]Wang, L., Anderson, K., Erwin, T., and Kapadia, S., "High-Order Methods for Solutions of Three-Dimensional Turbulent Flows," AIAA Paper 2013–0856, 2013.

[24]Wang, L. and Anderson, W. K., "Shape Sensitivity Analysis for the Compressible Navier-Stokes Equations via Discontinuous Galerkin Methods," *Computers and Fluids*, Vol. 69, No. 1, 2012, pp. 93–107.

[25]Wang, L., Anderson, W. K., Erwin, J. T., and Kapadia, S., "Discontinuous-Galerkin and Petrov-Galerkin Methods for Compressible Viscous Flows," *Computers and Fluids*, Vol. 100, No. 1, 2014, pp. 13–29.

[26]Venkatakrishnan, V., Allmaras, S. R., Johnson, F. T., and Kamenetskii, D. S., "Higher Order Schemes for the Compressible Navier-Stokes Equations," AIAA Paper 2003–3987, 2003.

[27]Hughes, T. J., Mazzei, L., and Jansen, K. E., "Large Eddy Simulation and the Variational Multiscale Method," *Computing and Visualization in Science*, Vol. 3, No. 1-2, 2000, pp. 47–59.

[28]Park, M. A. and Anderson, W. K., "Spatial Convergence of Three-Dimensional Turbulent Flows," AIAA Paper 2016–0859, 2016.

[29]Bazilevs, Y., Calo, V., Cottrell, J., Hughes, T., Reali, A., and Scovazzi, G., "Variational Multiscale Residual-Based Turbulence Modeling for Large Eddy Simulation of Incompressible Flows," *Computer Methods in Applied Mechanics and Engineering*, Vol. 197, No. 1, 2007, pp. 173–201.

[30]Glasby, R., Erwin, T., Stefanski, D. L., Allmaras, S., Galbraith, M. C., Anderson, W. K., and Nichols, R. H., "Introduction to COFFE: The Next-Generation HPCMP CREATE-AV CFD Solver," AIAA Paper 2016–0567, 2016.

[31]Whiting, C. H. and Jansen, K. E., "A Stabilized Finite-Element Method for the Incompressible Navier-Stokes Equations Using a Hierarchical Basis," *International Journal for Numerical Methods in Fluids*, Vol. 35, No. 1, 2001, pp. 93–116.

[32]Jansen, K. E., "A Stabilized Finite Element Method for Computing Turbulence," *Computer methods in applied mechanics and engineering*, Vol. 174, No. 3, 1999, pp. 299–317.

[33]Jansen, K., Johan, Z., and Hughes, T. J., "Implementation of a One-Equation Turbulence Model Within a Stabilized Finite Element Formulation of a Symmetric Advective-Diffusive System," *Computer methods in applied mechanics and engineering*, Vol. 105, No. 3, 1993, pp. 405–433.

[34]Kamenetskiy, D. S., Bussoletti, J. E., Hilmes, C. L., Venkatakrishnan, V., Wigton, L. B., and Johnson, F. T., "Numerical Evidence of Multiple Solutions for the Reynolds–Averaged Navier–Stokes Equations for High–Lift Configurations," AIAA Paper 2013–663, 2013.

[35]Anderson, W. K., Ahrabi, B. R., and Newman, J. C., "Finite-Element Solutions for Turbulent Flow over the NACA 0012 Airfoil," *AIAA Journal*, Vol. 54, No. 9, Sept. 2016, pp. 2688–2704.

[36]Newman, J. C. and Anderson, W. K., "Investigation of Unstructured Higher-Order Methods for Unsteady Flow and Moving Domains," AIAA Paper 2015-2917, 2015.

[37]Burgess, N. K. and Mavriplis, D. J., "*hp*-adaptive Discontinuous Galerkin Methods for the Navier-Stokes Equations," *AIAA Journal*, Vol. 50, No. 12, Dec 2012, pp. 2682–2694.

[38]Bonhaus, D. L., *A Higher Order Accurate Finite Element Method for Viscous Compressible Flows*, Ph.D. thesis, Virginia Polytechnic Institute and State University, 1998.

[39]Allmaras, S. R., Johnson, F. T., and Spalart, P. R., "Modifications and Clarifications for the Implementation of the Spalart-Allmaras Turbulence Model," *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, 2012.

[40]Spalart, P. R. and Allmaras, S. R., "A One-Equation Turbulence Model for Aerodynamic Flows," *La Recherche Aérospatiale*, Vol. 1, 1994, pp. 5–21.

[41]White, F. M. and Corfield, I., *Viscous fluid flow*, Vol. 3, McGraw-Hill New York, 2006.

[42]Barth, T. J., *Numerical Methods for Gasdynamic Systems on Unstructured Meshes*, Springer Berlin Heidelberg, Berlin, Heidelberg, 1999, pp. 195–285.

[43]Hartmann, R. and Leicht, T., "Generalized Adjoint Consistent Treatment of Wall Boundary Conditions for Compressible Flows," *Journal of Computational Physics*, Vol. 300, 2015, pp. 754–778.

[44]Galbraith, M. C. and Ollivier-Gooch, C., "BI2 - Smooth bump, BL1 - Laminar airfoil, and BR1 - Turbulent airfoilnic," *Invited Presentation at the 4th International Workshop on High-Order CFD Methods)*, Eccomas / 6th European Conference on CFD (ECFD VI), 2016.

[45]Ahrabi, B. R., *An hp-Adaptive Petrov-Galerkin Method for Steady-State and Unsteady Problems*, Ph.D. thesis, University of Tennessee at Chattanooga, Aug. 2015.

[46]Allmaras, S., "Lagrange Multiplier Implementation of Dirichlet Boundary Conditions in Compressible Navier–Stokes Finite-Element Methods," Aiaa paper, 2005.

[47]Pandya, M. J., Diskin, B., Thomas, J. L., and Frink, N. T., "Improved Convergence and Robustness of USM3D Solutions on Mixed Element Grids," AIAA Paper 2015–1747, 2013.

[48]Burgess, N. and Glasby, R., "Advances in Numerical Methods for CREATE-AV Analysis Tools," AIAA Paper 2014–417, 2014.

[49]Saad, Y. and Schultz, M. H., "GMRES: A Generalized Minimum Residual Algorithm for Solving Nonsymmetric Linear Systems," *SIAM Journal of Scientific and Statistical Computing*, Vol. 7, 1986, pp. 856–869.

[50]Saad, Y., *Iterative Methods for Sparse Linear Systems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2nd ed., 2003.

[51]de Barros Ceze, M. A., *A Robust hp-Adaptation Method for Discontinuous Galerkin Discretizations Applied to Aerodynamic Flows*, Ph.D. thesis, University of Toronto, 2013.

[52]Wrenn, G. A., "An Indirect Method for Numerical Optimization Using the Kreisselmeier-Steinhauser Function," NASA CR-4220, Langley Research Center, 1989.

[53]von Neumann, J. and Richtmyer, R. D., "A Method for the Numerical Calculation of Hydrodynamic Shocks," *Journal Applied Physics*, Vol. 21, No. 3, March 1950, pp. 232–237.

[54]Persson, P.-O. and Peraire, J., "Sub-Cell Shock Capturing for Discontinuous Galerkin Methods," AIAA Paper 2006–112, 2006.

[55]Liepmann, H. and Roshko, A., *Elements of Gasdynamics*, Dover Books on Aeronautical Engineering Series, Dover Publications, 1957.

[56]Larsson, J., Vicquelin, R., and Bermejo-Moreno, I., "Large eddy simulations of the HyShot II scramjet," *Center for Turbulence Research Annual Research Briefs*, 2011, pp. 63–74.

[57]Oberkampf, W. L. and Roy, C. J., *Verification and Validation in Scientific Computing*, Cambridge University Press, 2010.

[58]Erwin, J. T., *Stabilized Finite Elements for Compressible Turbulent Navier Stokes*, Ph.D. thesis, University of Tennessee at Chattanooga, Dec. 2013.

[59]Liu, C., *A Stabilized Finite Element Dynamic Overset Method for the Navier-Stokes Equations*, Ph.D. thesis, University of Tennessee at Chattanooga, May 2016.

[60]Rajamohan, S., *A Streamline Upwind/Petrov-Galerkin FEM Based Time-Accurate Solution of 3D Time-Domain Maxwell's Equations for Dispersive Materials*, Ph.D. thesis, University of Tennessee at Chattanooga, Aug. 2014.

[61]Zhang, X., *Higher-Order Petrov-Galerkin Methods for Analysis of Antennas*, Master's thesis, University of Tennessee at Chattanooga, Oct. 2013.

[62]Shoemake, W. L., *Extension of a High-Order Petrov-Galerkin Implementation Applied to Non-Radiating and Radar Cross Section Geometries*, Master's thesis, University of Tennessee at Chattanooga, Dec. 2013.

[63]Lin, W., *Design Optimization of Acoustic Metamaterials and Phononic Crystals with a Time Domain Method*, Ph.D. thesis, University of Tennessee at Chattanooga, Dec. 2016.

[64]Rumsey, C. L., Smith, B., and Huang, G. P., "Description of a Website Resource for Turbulence Model Verification and Validation," AIAA Paper 2010–4742, 2010.

[65]Gnoffo, P. A. and White, J. A., "Computational Aerothermodynamic Simulation Issues on Unstructured Grids," AIAA Paper 2004–2371, 2004.

[66]Gnoffo, P. A., "Simulation of Stagnation Region Heating in Hypersonic Flow on Tetrahedral Grids," AIAA Paper 2007-3960, 2007.

[67]Vassberg, J. C., Tinocoand, E., Mani, M., Brodersen, O., Eisfeld, B., Wahls, R., Morrison, J. H., Zickuhr, T., Laflin, K., and Mavriplis, D., "Summary of the Third AIAA CFD Drag Prediction Workshop," AIAA Paper 2007–260, 2007.

[68]Morrison, J. H. and Hemsch, M. J., "Statistical Analysis of CFD Solutions from the Third AIAA Drag Prediction Workshop," AIAA Paper 2007–254, 2007.

[69]Gnoffo, P. A., "Multi-dimensional, Inviscid Flux Reconstruction for Simulation of Hypersonic Heating on Tetrahedral Grids," AIAA paper 2009-599, 2009.

[70]Gnoffo, P. A., "Updates to Multi-dimensional Flux Reconstruction for Hypersonic Simulations on Tetrahedral Grids," AIAA Paper 2010-1271, 2010.

[71]Edwards, J. R., "A Low-Diffusion Flux-Splitting Scheme for Navier-Stokes Calculations," *Computers & Fluids*, Vol. 26, No. 6, 1997, pp. 635–659.

[72]Van Albada, G., Van Leer, B., and Roberts Jr, W., "A Comparative Study of Computational Methods in Cosmic Gas Dynamics," *Upwind and High-Resolution Schemes*, Springer, 1997, pp. 95–103.

[73]Schmitt, V. and Charpin, F., "Pressure Distributions on the ONERA-M6-wing at Transonic Mach Numbers," *Experimental data base for computer program assessment*, Vol. 4, 1979.